A rundown of recent contributions to STEAM

Andrea Vitrano

Machine Protection and Electrical Integrity Group (TE-MPE-PE)

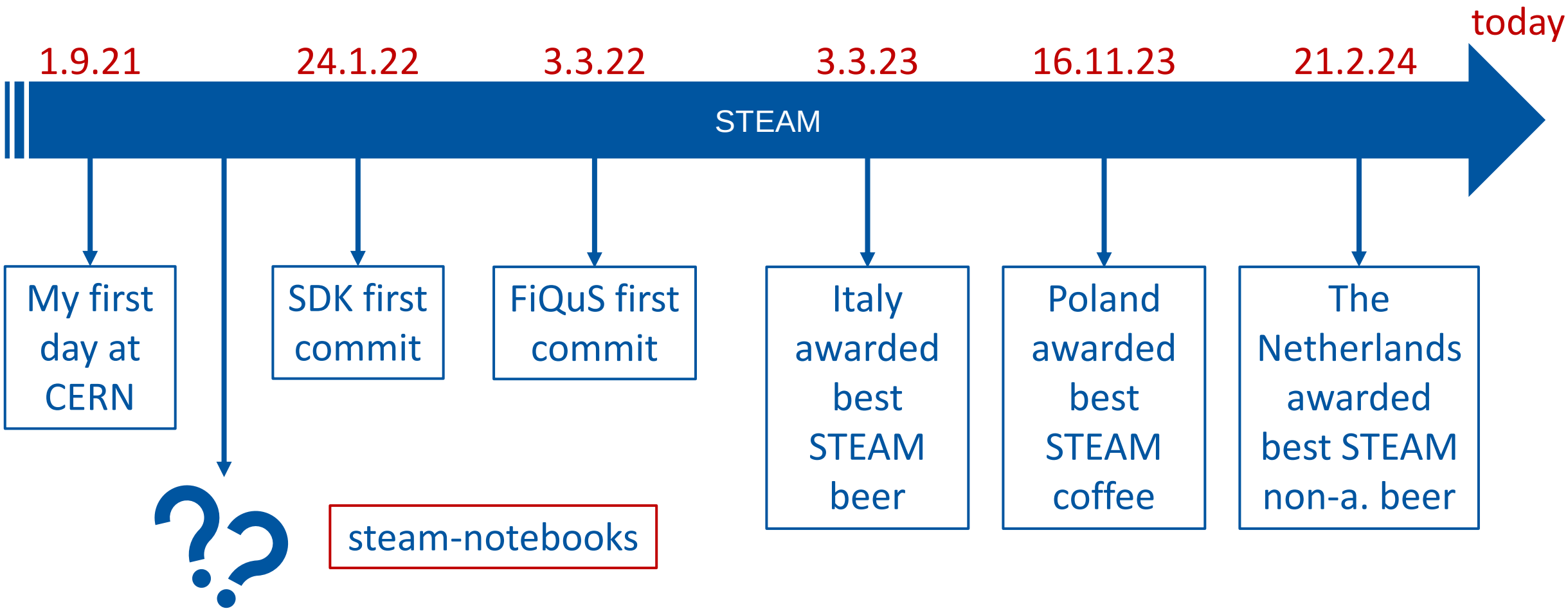
24th July 2024 – PE section meeting

cern.ch/steam

Outline

1. Pre-STEAM-SDK Era
2. STEAM-SDK Contributions
3. FiQuS Contributions
4. Conclusions

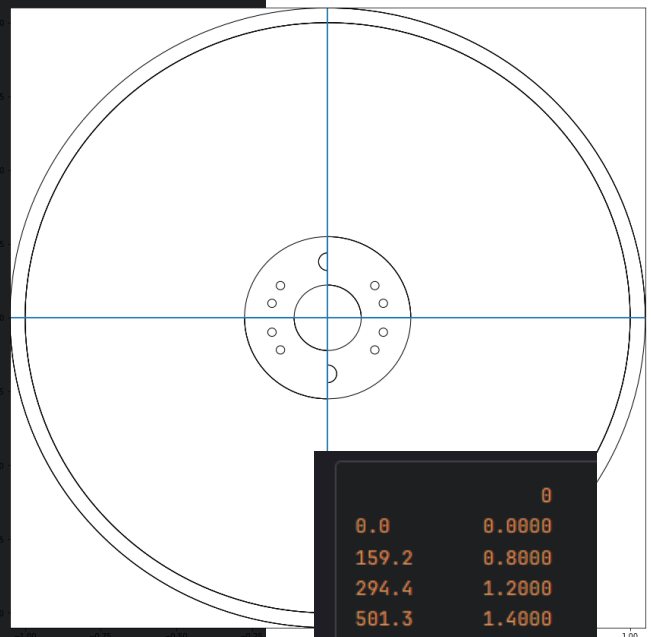
Timeline


today
1.9.21
24.1.22
3.3.22
3.3.23
16.11.23
21.2.24
STEAM
My first day at CERN
SDK first commit
FiQuS first commit
Italy awarded best STEAM beer
Poland awarded best STEAM coffee
The Netherlands awarded best STEAM non-a. beer
??
steam-notebooks

From notebooks to automatized model generation

```
# iron yoke holes
mm = 0.001

x1 = 0 * mm
y1 = 190 * mm
r1 = 30 * mm

x2 = 156 * mm
y2 = 109 * mm
r2 = 14 * mm

x3 = 184 * mm
y3 = 49 * mm
r3 = 14 * mm

kp1 = Point.ofCartesian(x1, y1)
kp2 = Point.ofCartesian(x2, y2)
kp3 = Point.ofCartesian(x3, y3)

kp1_1 = Point.ofCartesian(x1, y1 + r1)
kp1_3 = Point.ofCartesian(x1 + r1, y1)
kp1_2 = Point.ofCartesian(x1, y1 - r1)

ln1_1 = Arc.ofThreePoints(kp1_1, kp1_3, kp1_2)
ln1_2 = Line.ofEndPoints(kp1_1, kp1_2)

ln2 = Circumference.ofCenterRadius(kp2, r2)
ln3 = Circumference.ofCenterRadius(kp3, r3)

ar1 = Area.ofHyperLines(a.create_hyper_line_array(gateway, (ln1_1, ln1_2)))
ar2 = Area.ofHyperLines(a.create_hyper_line_array(gateway, (ln2)))
ar3 = Area.ofHyperLines(a.create_hyper_line_array(gateway, (ln3)))

ar1_2 = ar1.mirrorY()
ar1_3 = ar1_2.mirrorX()
ar1_4 = ar1.mirrorX()

ar2_2 = ar2.mirrorY()
ar2_3 = ar2_2.mirrorX()
ar2_4 = ar2.mirrorX()
```

Old STEAM-SIGMA



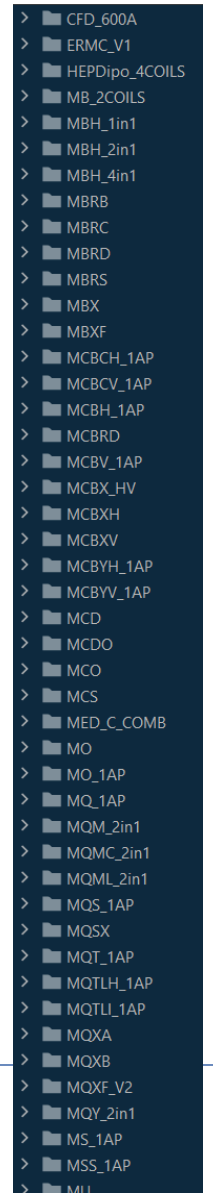
	0
0.0	0.0000
159.2	0.8000
294.4	1.2000
501.3	1.4000
795.8	1.5000
1154.0	1.5500
1795.0	1.6000
2862.0	1.6500
4383.0	1.7000
6044.0	1.7500
8122.0	1.8000
10590.0	1.8500
13160.0	1.9000
21170.0	2.0000
26750.0	2.0500

Difficult in Jupyter Notebook to do:

- Modular and re-usable code
- Version control
- Debugging

Steam-notebooks:

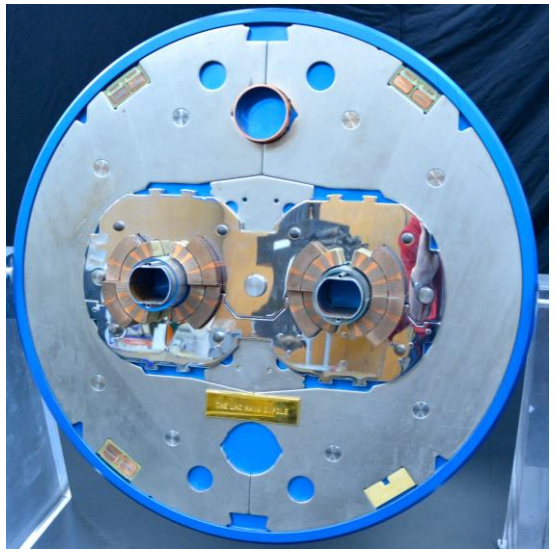
- Magnet-specific scripts
- Hard-coded data
- Redundant code



The ROXIE parser

```
kpi8=[B-0.005,-0.000001+A+0.01];
kpi9=[B,A+0.01];
kpi10=[B,Sqrt(od*od-B*B)];

kpi11=[Sqrt(od*od-B*B),B];
kpi12=[A+0.01,B];
kpi13=[-0.000001+A+0.01,B-0.005];
kpi14=[0.000001+A,B-0.005];
kpi15=[A,Sqrt(od*od-A*A)];

kpi16=[0.015,Sqrt(od*od-0.015*0.015)];
kpi17=[0.015,Sqrt(od*od-0.015*0.015)-0.0104];
kpi18=[0,Sqrt(od*od-0.015*0.015)-0.0104];

--HX-hole(f60_R227.5)
kpi201=[rhx/Sqrt(2.0)+dhx/2.0,rhx/Sqrt(2.0)];
kpi202=[rhx/Sqrt(2.0)-dhx/2.0,rhx/Sqrt(2.0)];

kpi101=[0,0.126];
kpi102=[0.00505,0.126];

kpi21=[0.00505,0.117-0.001705];
kpi22=[0.03,0.117-0.030/Tan(71.3405/180*Pi)];
kpi23=[0.111,0];

--IRON LINE;
lni5=HyperLine(kpi6,kpi82,"Arc",od,0.5);
lni6=HyperLine(kpi6,kpi7,"Line",0.5);
lni7=HyperLine(kpi7,kpi8,"Arc",0.005,0.5);
lni8=HyperLine(kpi8,kpi9,"Line",0.5);
lni9=HyperLine(kpi9,kpi10,"Line",0.5);
lni10=HyperLine(kpi11,kpi10,"Arc",od,0.5);
lni11=HyperLine(kpi11,kpi12,"Line",0.5);
```

ROXIE .iron file



.iron parser

- Iron yoke
- Steel collars
- Keys

```yaml
Roxie_Data:
  iron:
    key_points:
    hyper_lines:
    hyper_areas:
      aryoke:
        material: BHiron2
        lines: [lnyoke_1, lnyoke_2b, lnyo
      arbarI:
      arbarII:
      arbarIII:
      arh1:
      arh2:
      arh3:
      arcollar_r:
      arcollar_l:
      arhc1:
      arhc2:
    hyper_holes:
      1:
        areas: [arh1, aryoke]
      2:
      3:
```

The Pydantic library and data validation

Pydantic

- Data validation
- Python type annotations
- Serialization to YAML format
- JSON Schema compatible

Pydantic class example

```python
class MultipoleGeometry(BaseModel):
    """
    Level 2: Class for FiQuS Multipole
    """
    geom_file_path: Optional[str] = Field(
        default=None,
        description="It contains the path to a .geom file. If null, the default
    )
    plot_preview: Optional[bool] = Field(
        default=False,
        description="If true, it displays matplotlib figures of the magnet geome
    )
    electromagnetics: MultipoleGeometryElectromagnetics = Field(
        default=MultipoleGeometryElectromagnetics(),
        description="This dictionary contains the geometry information for the e
    )
    thermal: MultipoleGeometryThermal = Field(
        default=MultipoleGeometryThermal(),
        description="This dictionary contains the geometry information for the t
    )
```

First usage with ROXIE data

↓

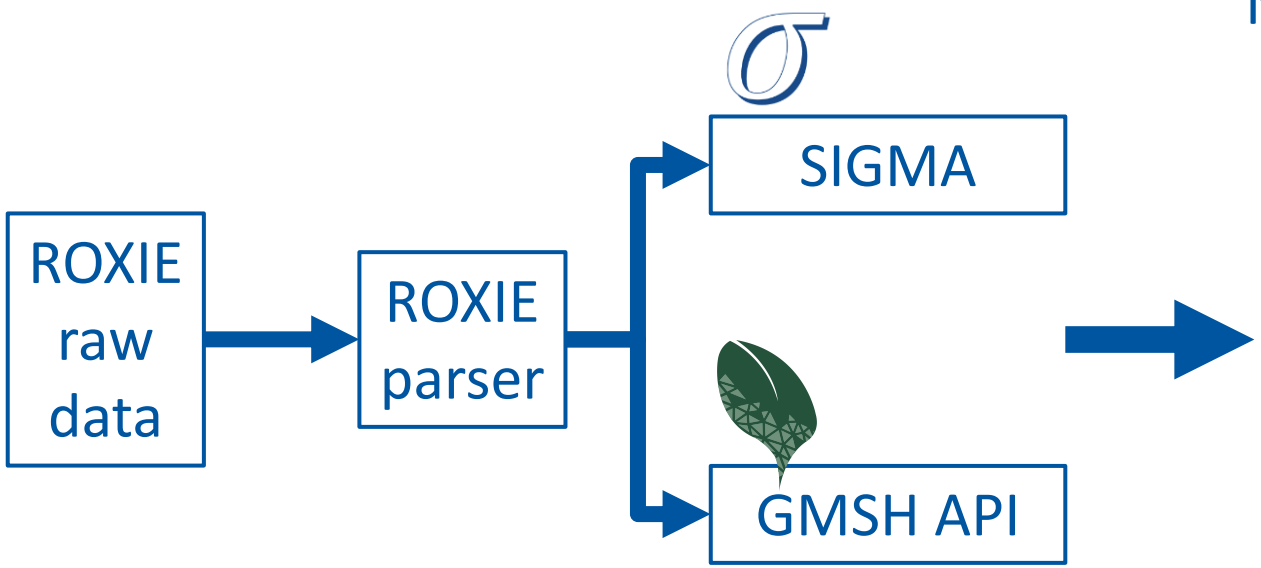
Converting ROXIE parser to use Pydantic objects

↓

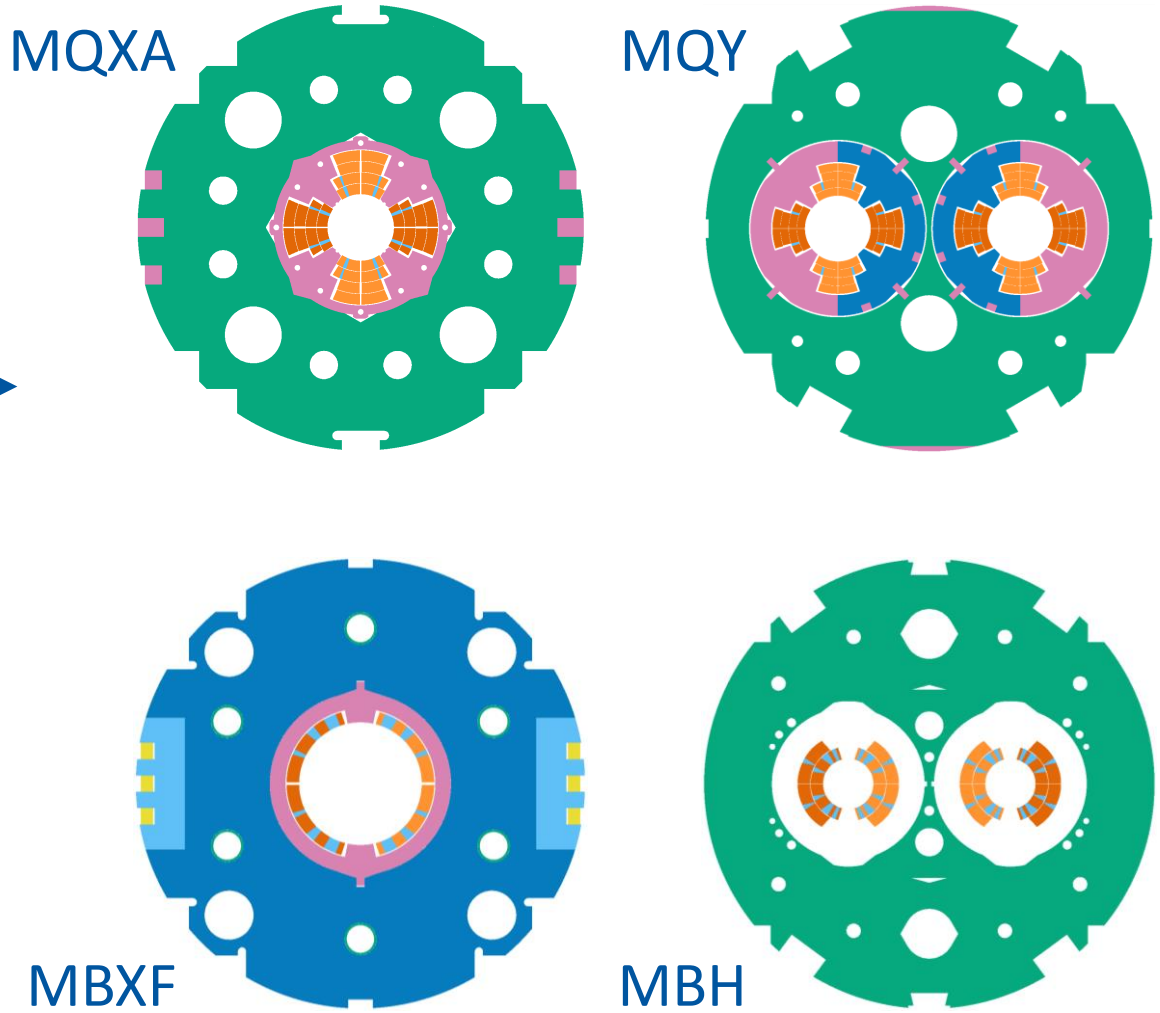
Standardized i/o

Many thanks to Mariusz Wozniak

Generalized geometry builder for multi-pole magnets



- Concise and maintainable code
- Scalable and re-usable code
- Automatized generation of models



The making of the modelData magnet library

	LEDET		SIGMA		BBQ
Sources	Coil geometry		Coil geometry	Source file name	
Sources			Icon yoke geometry	Source file name	
Sources			BH-curves	Source file name	
Sources	Wedges geometry		Wedges geometry	Source file name	
Sources	Self-mutual inductance	Source file name			
Sources	Magnetic field	Source file name			
General	magnet_name				
General	circuit_name				
General	model_version				
General	model_case				
General	model_state				
General	T00			Top (Operation temperature of cable)	
General	I_magnet			magLength (Magnetic length of magnet)	
Winding	GroupToCoilSection				
Winding	multipole	alphasDEG			
Winding	multipole	rotation_block			
Winding	multipole	rotation_block			
Winding	multipole	mirror_block			
Winding	multipole	mirrorY_block			
Winding	solenoid	...			
Winding	solenoid	...			
Winding	solenoid	...			
Winding	solenoid	...			
Winding	solenoid	...			
Winding	CCT	...			
Winding	CCT	...			
Winding	CCT	...			
Winding	CCT	...			
Winding	Electrical order	elPairs_GroupTogether			
Winding	Electrical order	elPairs_RevElOrder			
Winding	Heat exchange	max_distance			
Conductor	type				
Conductor	type to group				
Conductor	hIns_inGroup		geometry	wInsulNarrow (Thickness of insulation along height of cable)	
Conductor	wIns_inGroup		geometry	wInsulWide (Thickness of insulation along width of cable)	
Conductor	df_inGroup		geometry	dFilament (Diameter of filament in strand)	
Conductor	ds_inGroup		geometry	dstrand (Diameter of strand in cable)	
Conductor	Lp_f_inGroup		geometry	lTp (Filament twist pitch)	
Conductor	wBare_inGroup		geometry	wBare (Width of bare cable)	
Conductor	hBare_inGroup		geometry		
Conductor			geometry	hInBare (Smaller height of cable)	
Conductor			geometry	hOutBare (Larger height of cable)	

Relic of the past

Model Data

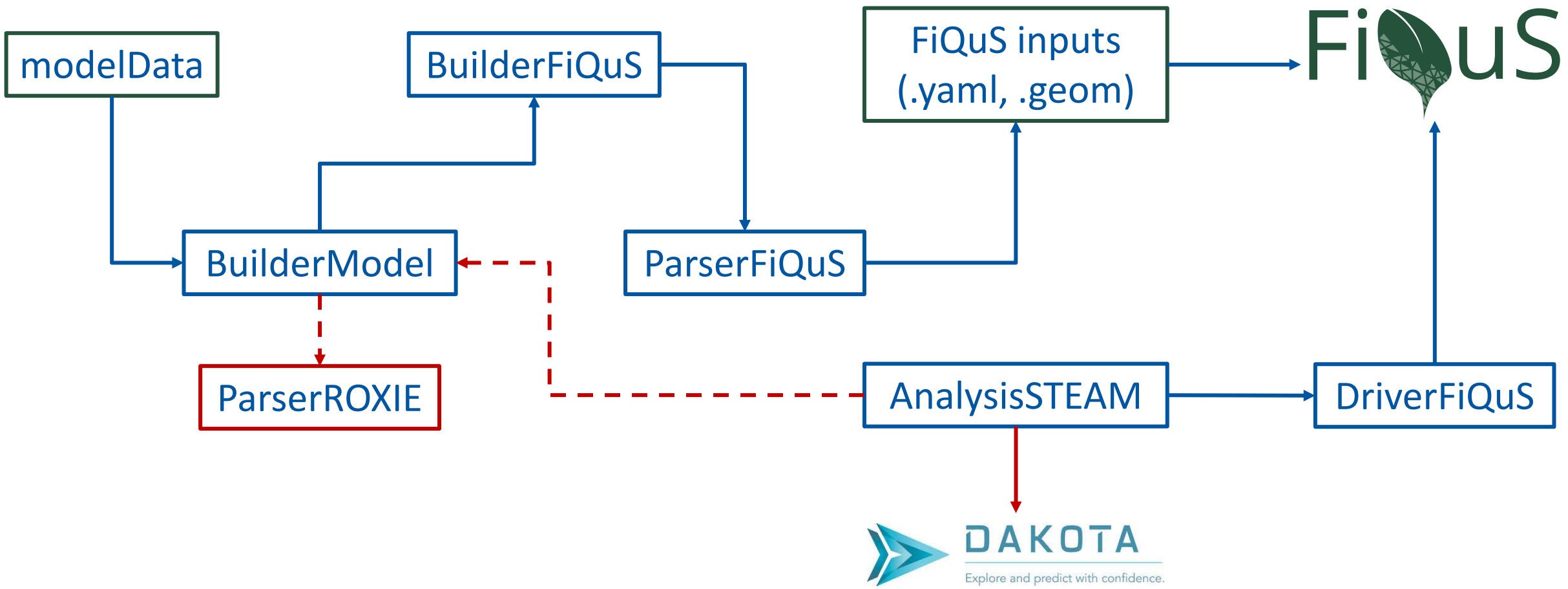
```
Sources:
  coil_fromROXIE: MQXA.data  # .data file
  conductor_fromROXIE: ../../roxie.cadata
  iron_fromROXIE: MQXA_2D_modified.iron
  BH_fromROXIE: ../../roxie.bhdata
  magnetic_field_fromROXIE: MQXA.map2d
  sm_inductance: null
GeneralParameters:
  magnet_name: MQXA
  circuit_name: null
  model:
    name: MQXA_reference
    version: v1.0
    case: null
    state: validated
  magnet_type: multipole
  T_initial: 1.9
  magnetic_length: 6.37
CoilWindings:
  conductor_to_group: [1, 1, 2, 1, 2, 2, 1, 1,
  group_to_coil_section: [1, 1, 1, 1, 1, 1, 1,
  polarities_in_group: [1, 1, 1, 1, 1, 1, -1, -
  n_half_turn_in_group: []
  half_turn_length: [6.37, 6.37, 6.37, 6.37, 6.
  electrical_pairs:
    group_together: [[1, 25], [2, 26], [4, 28],
    reversed: [1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0,
    overwrite_electrical_order: []
  multipole:
    wedges:
      material: Cu
      RRR: 179.8259386
      T_ref_RRR_high: 273.0
  pancake:
    tbc: null
  solenoid:
    coils: [{name: null, a1: null, a2: null, b1
  CCT_straight:
    winding_order: null
    winding_numberTurnsFormers: null
```

- Categorizing input parameters
- Grouping and generalizing

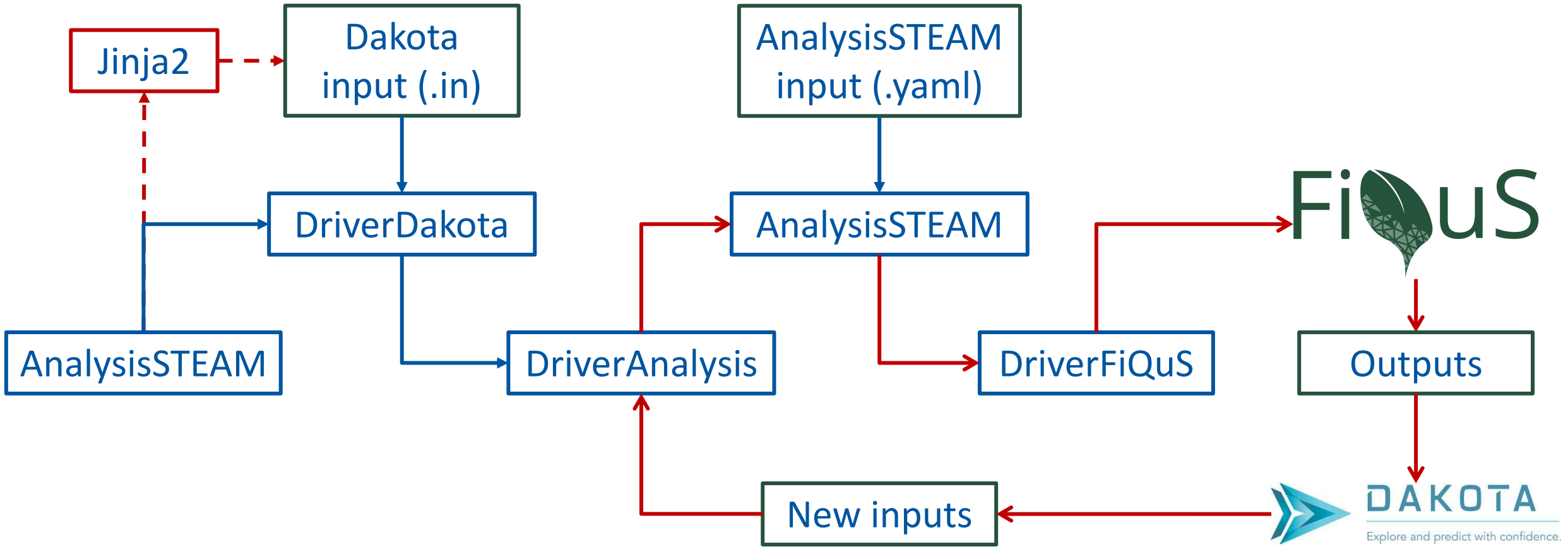
Unified magnet database across tools

Many thanks to Mariusz Wozniak and Emmanuele Ravaioli

The STEAM-SDK pipeline for FiQuS

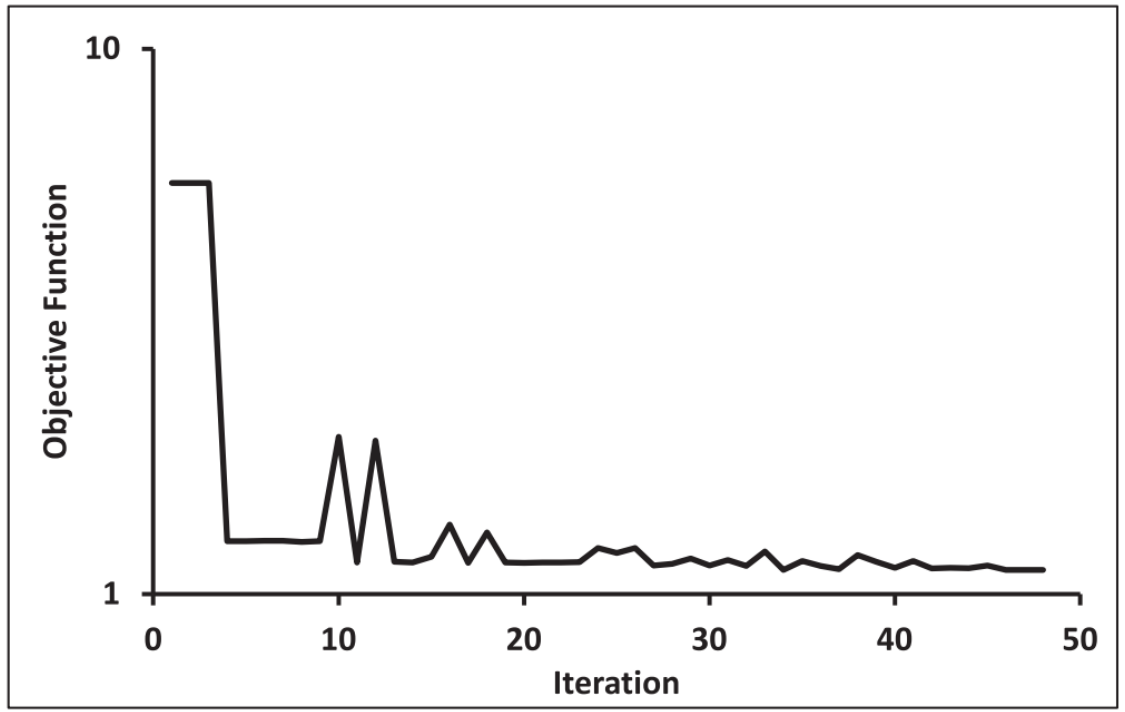


The integration of Dakota in STEAM-SDK

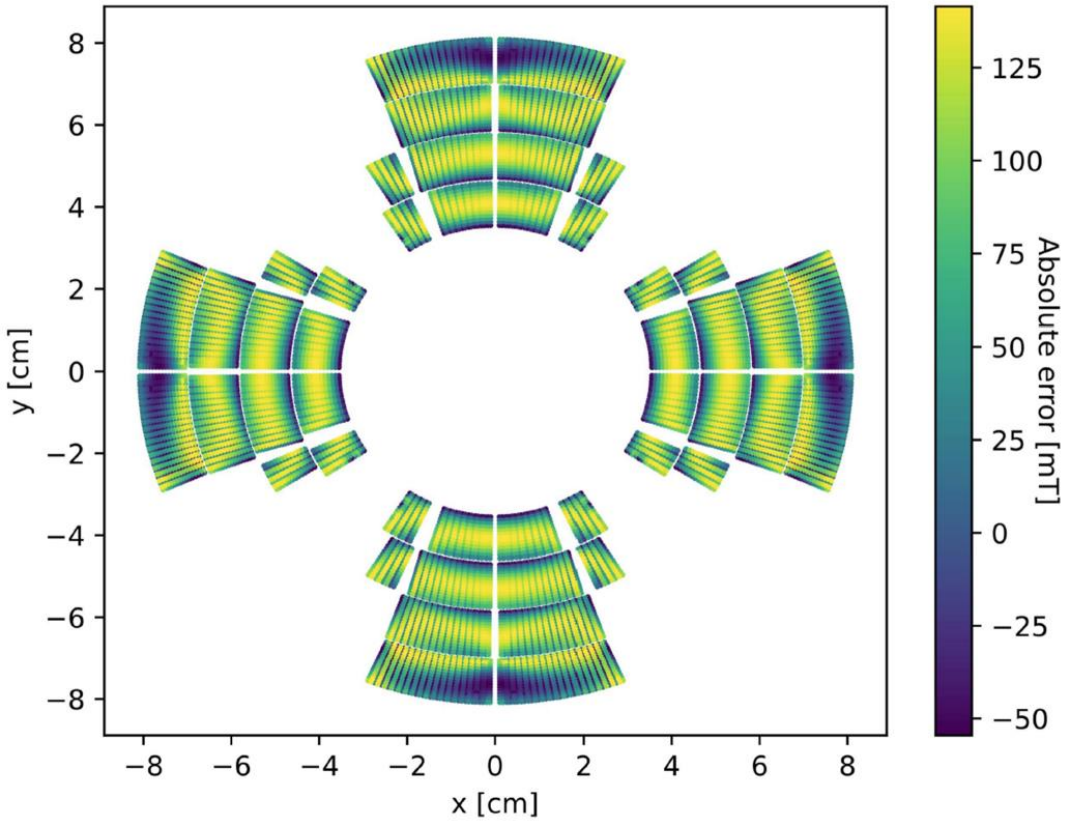


Many thanks to Mariusz Wozniak

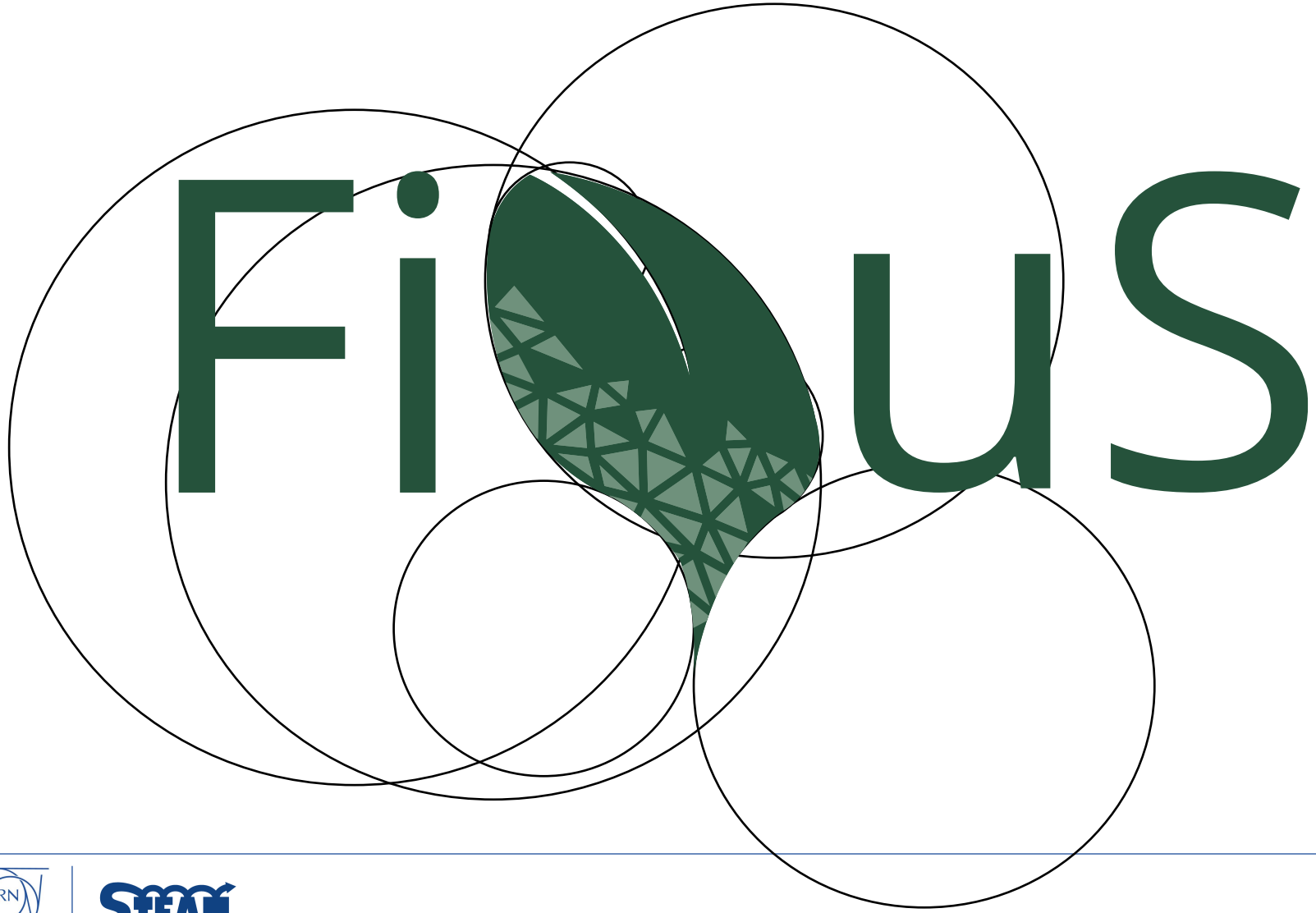
Dakota multi-objective optimization



FiQuS vs ROXIE (magnetic flux density)

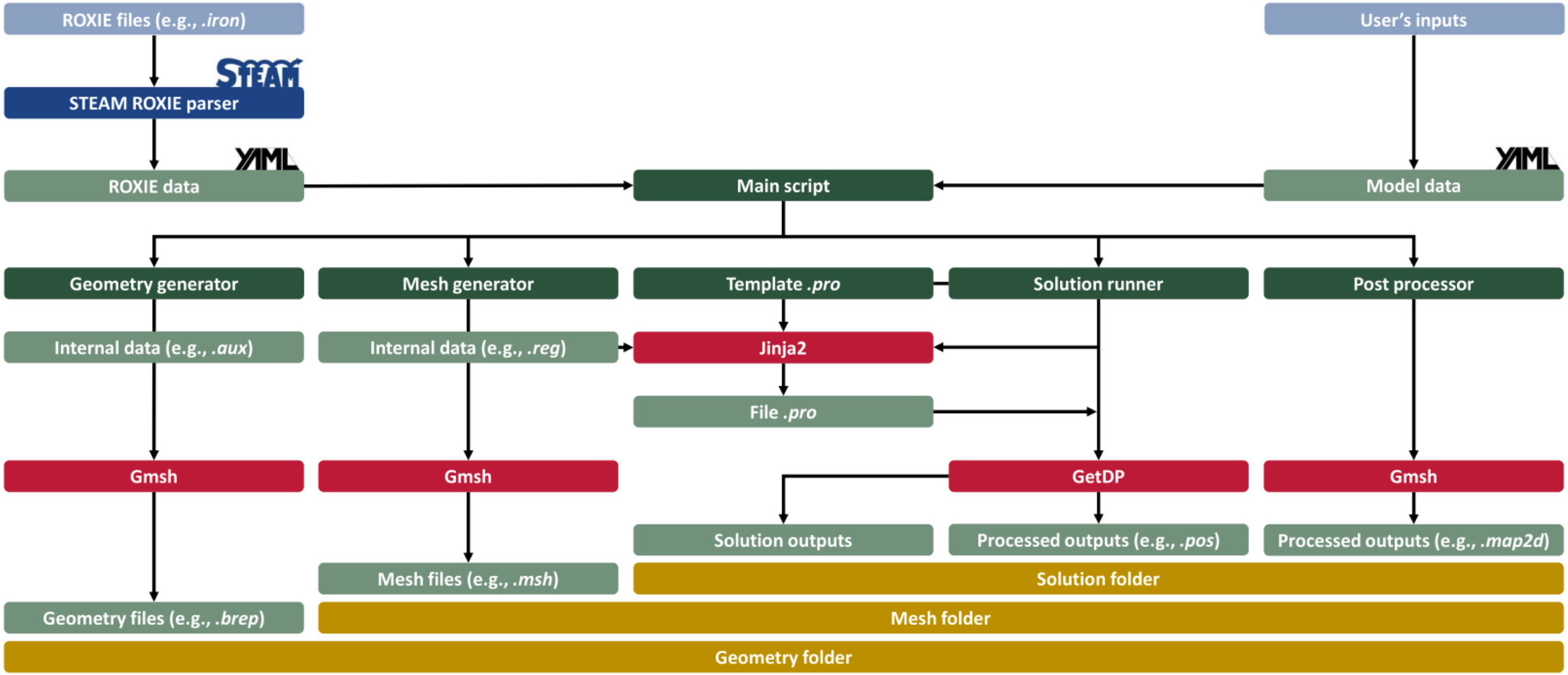


The making of the FiQuS logo

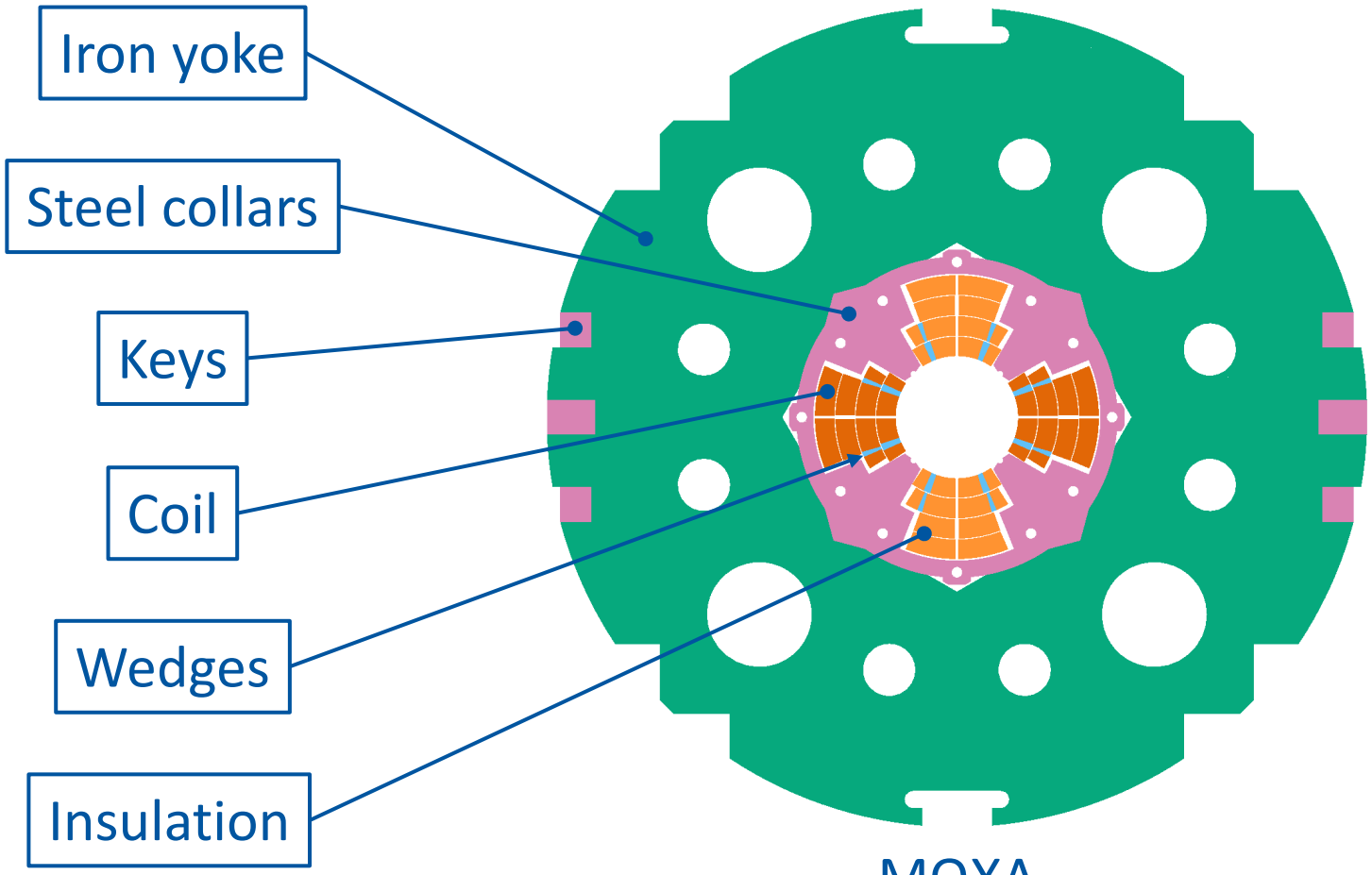


- Inkscape
- Weeping leaf
- Triangular mesh
- Golden ratio rule
- Dark green: 25523b
  Light green: 6f917c

FiQuS multi-pole code structure



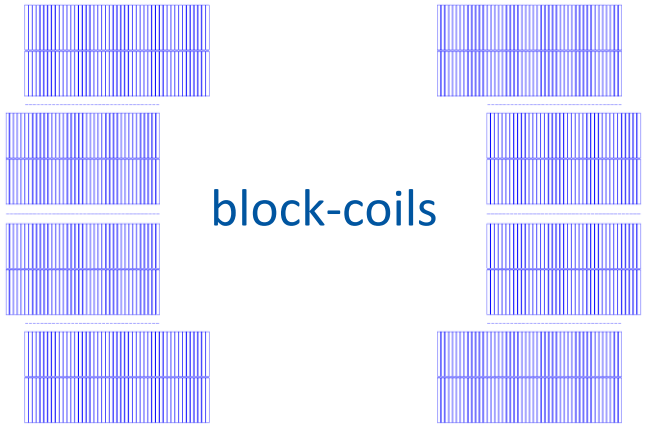
FiQuS multi-pole magnets



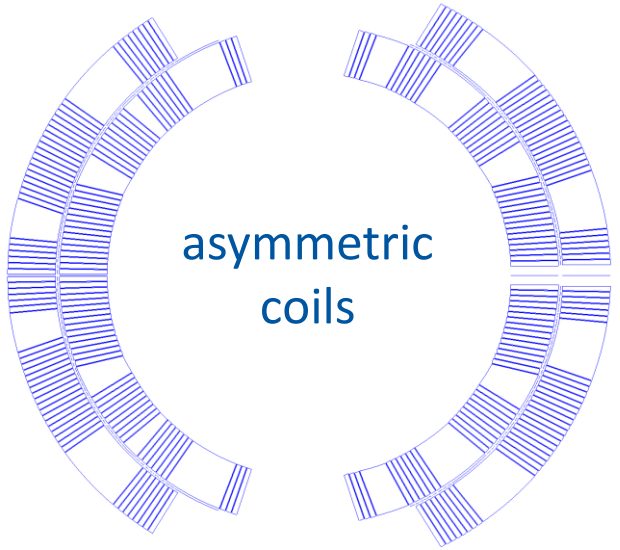
MQXA

- 2D multi-region geometries
- Aimed to EM-TH coupled transients for quench simulations

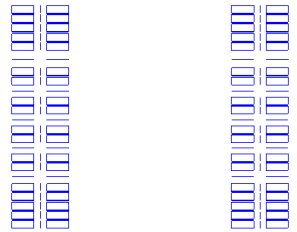
“Multiple” multi-pole magnets



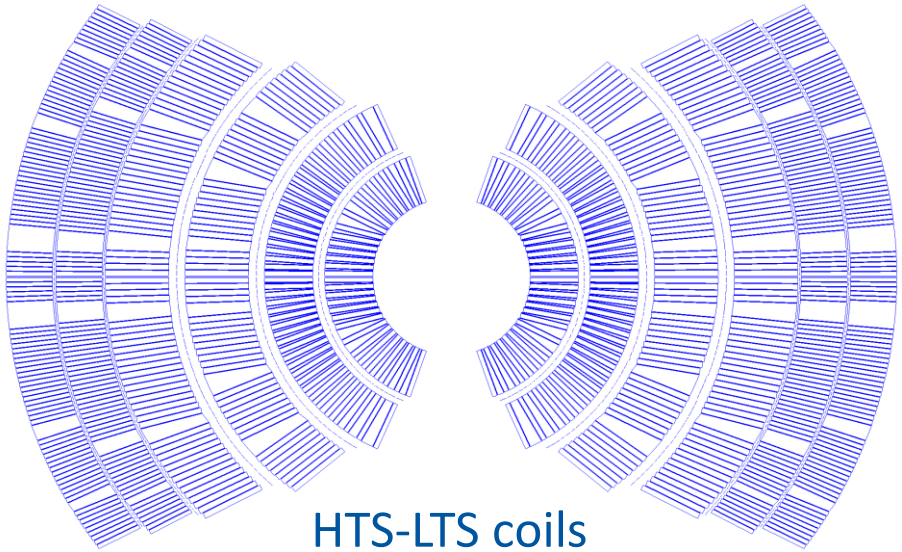
block-coils



asymmetric coils

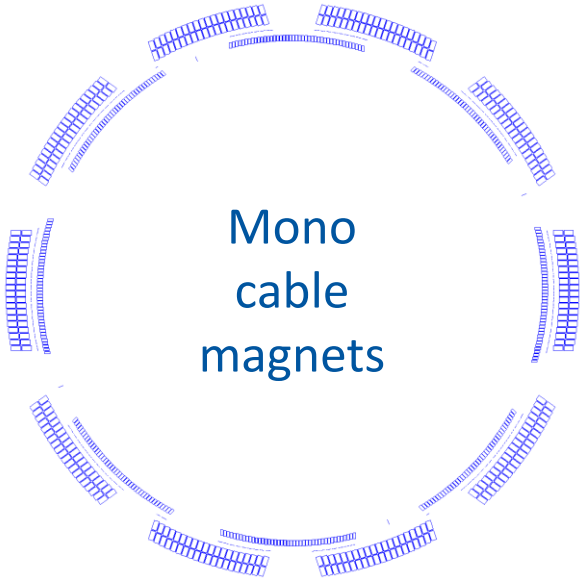


common-coils

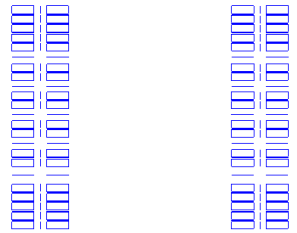


HTS-LTS coils

Cos-theta coils

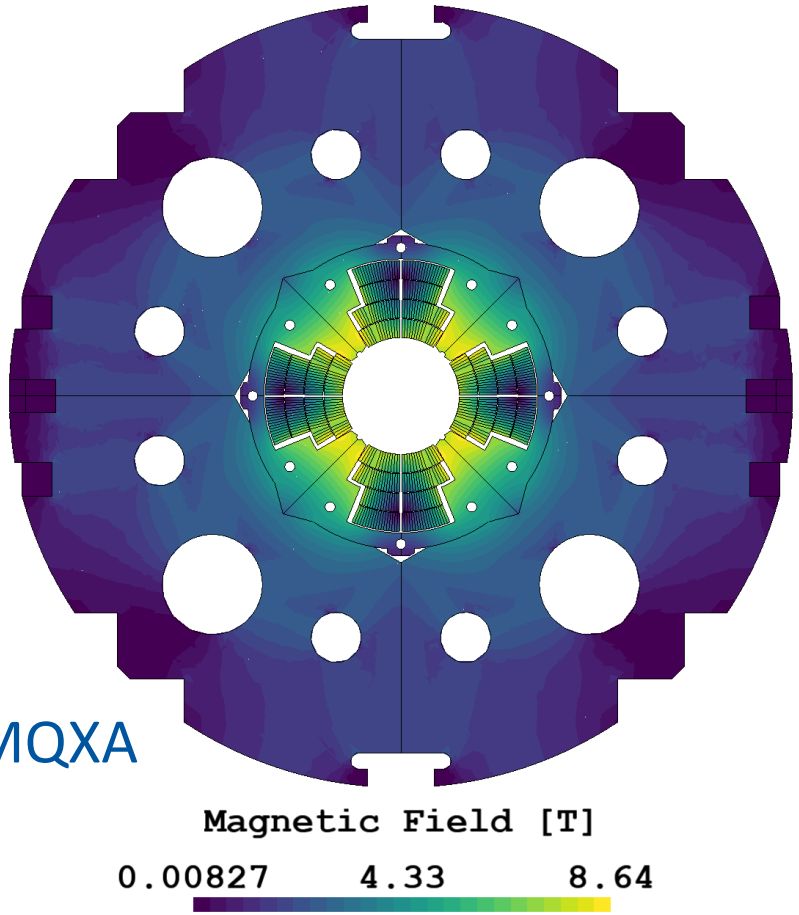


Mono cable magnets

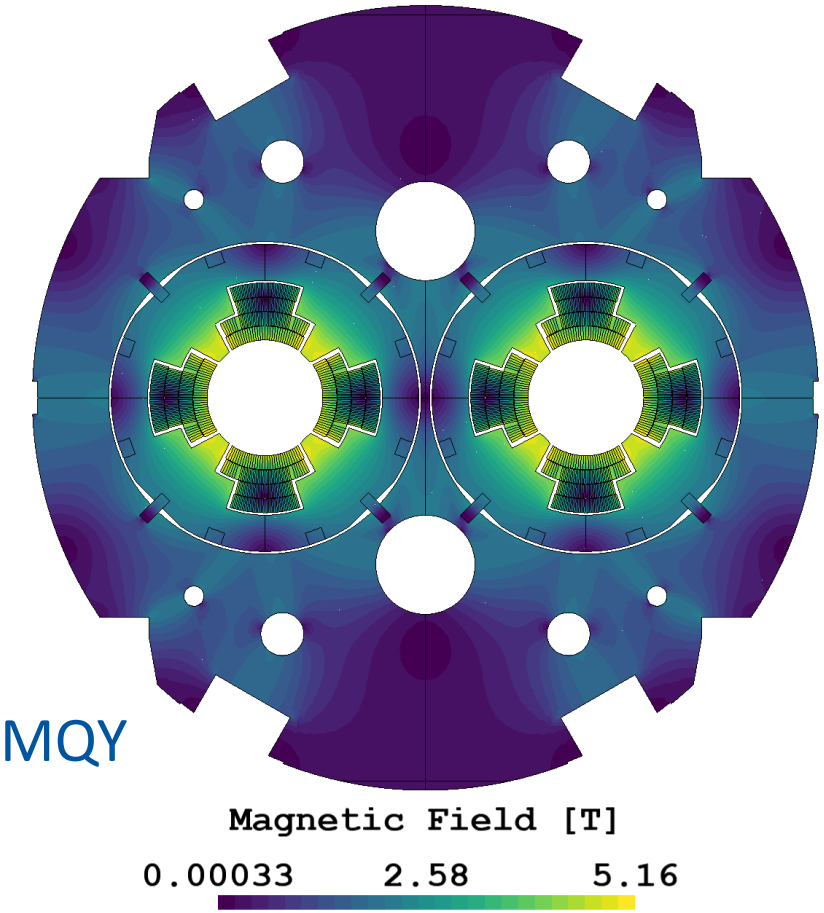


Magnetostatic simulations

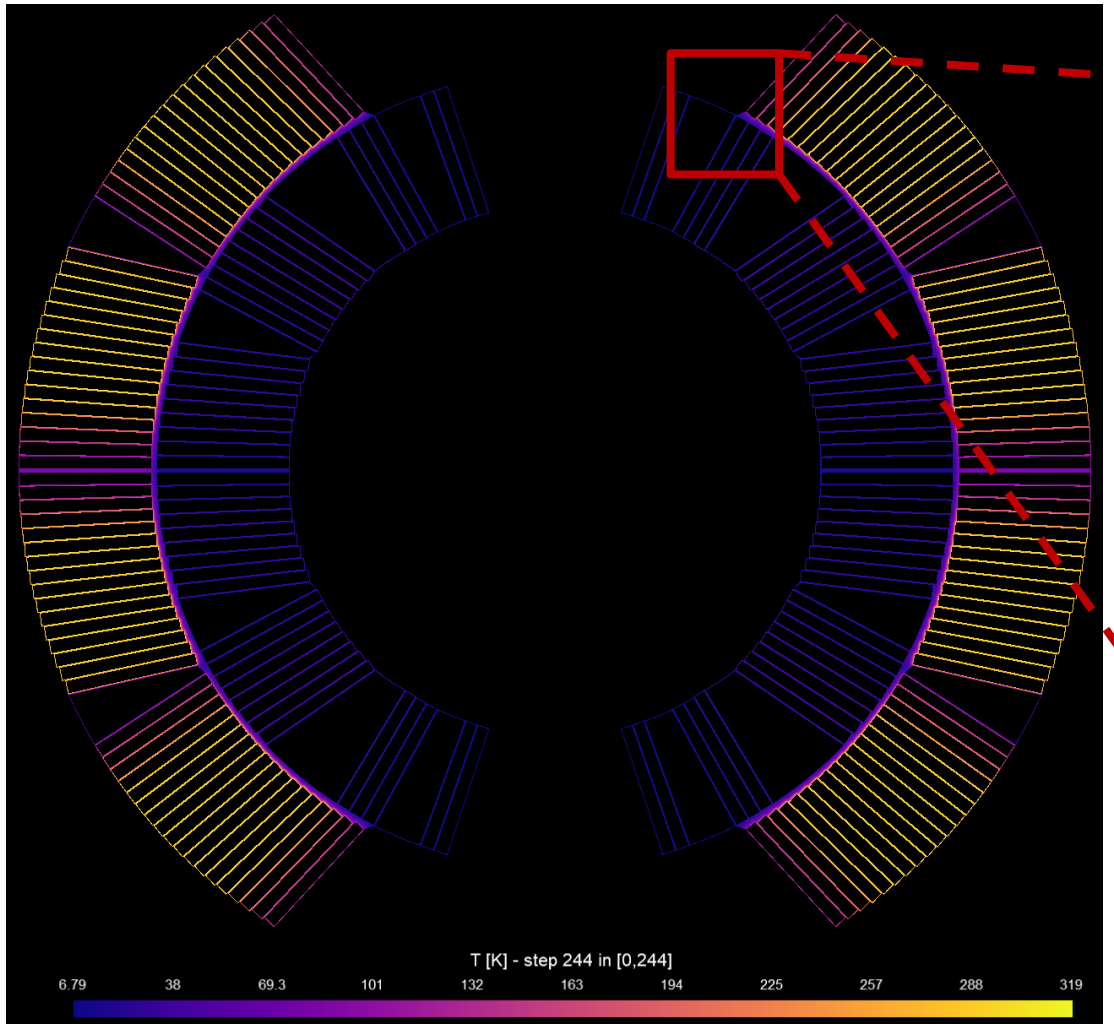
MQXA

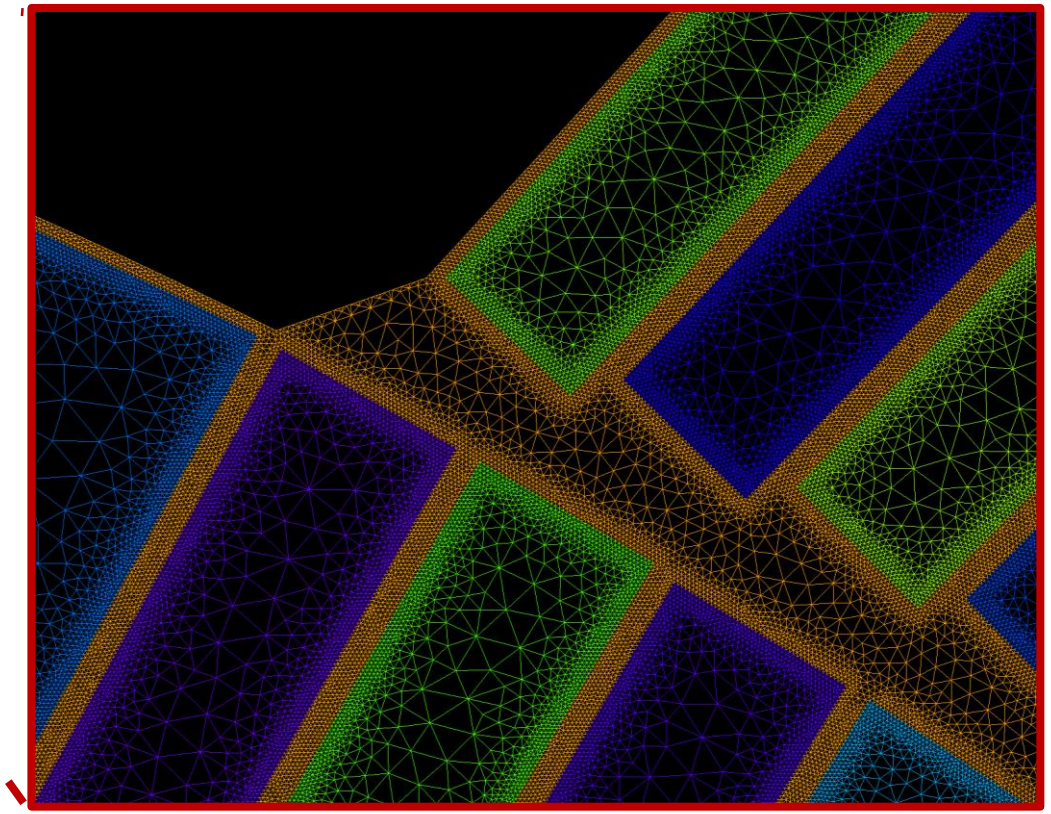


MQY



Thermal transient solution

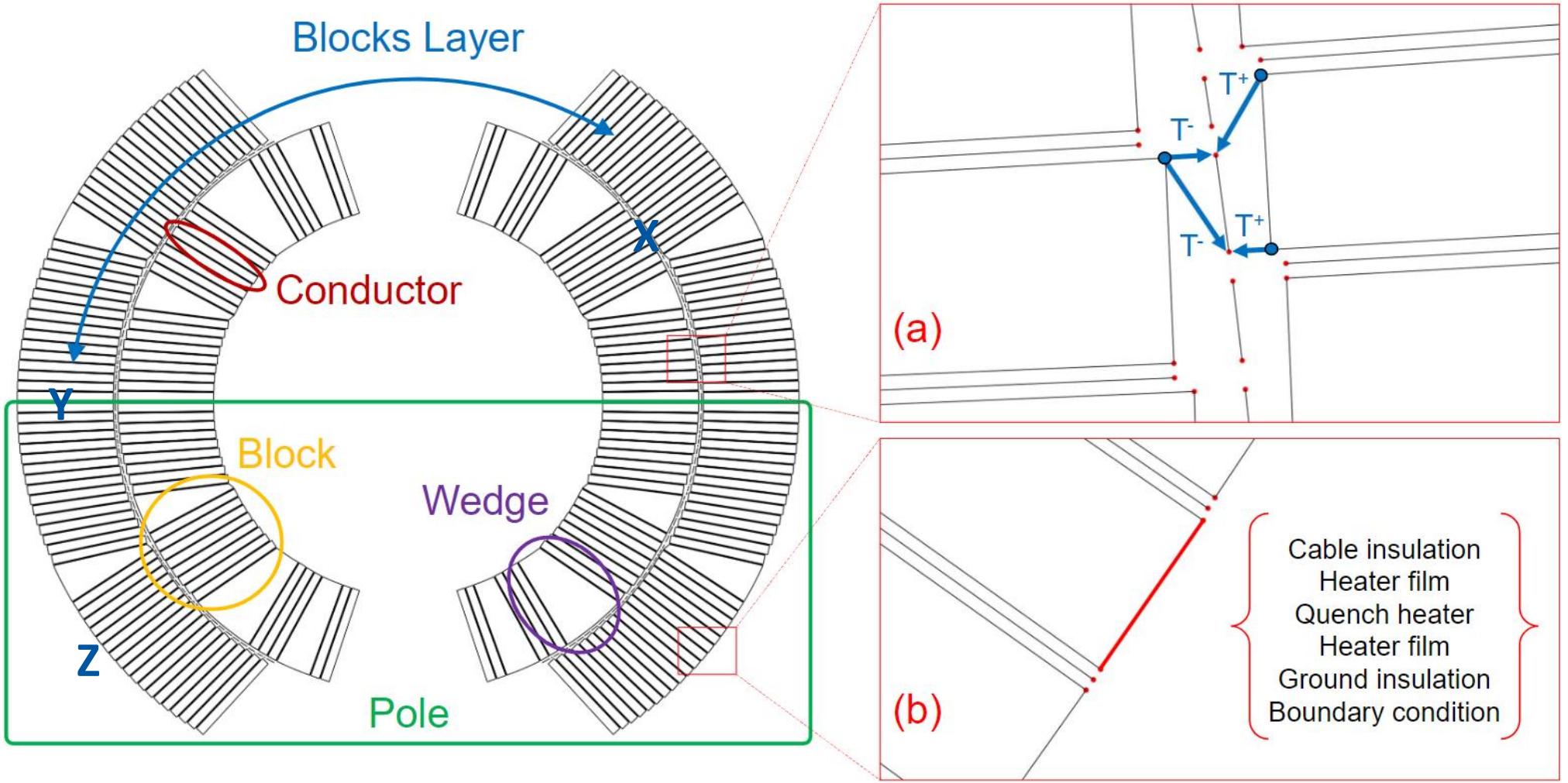




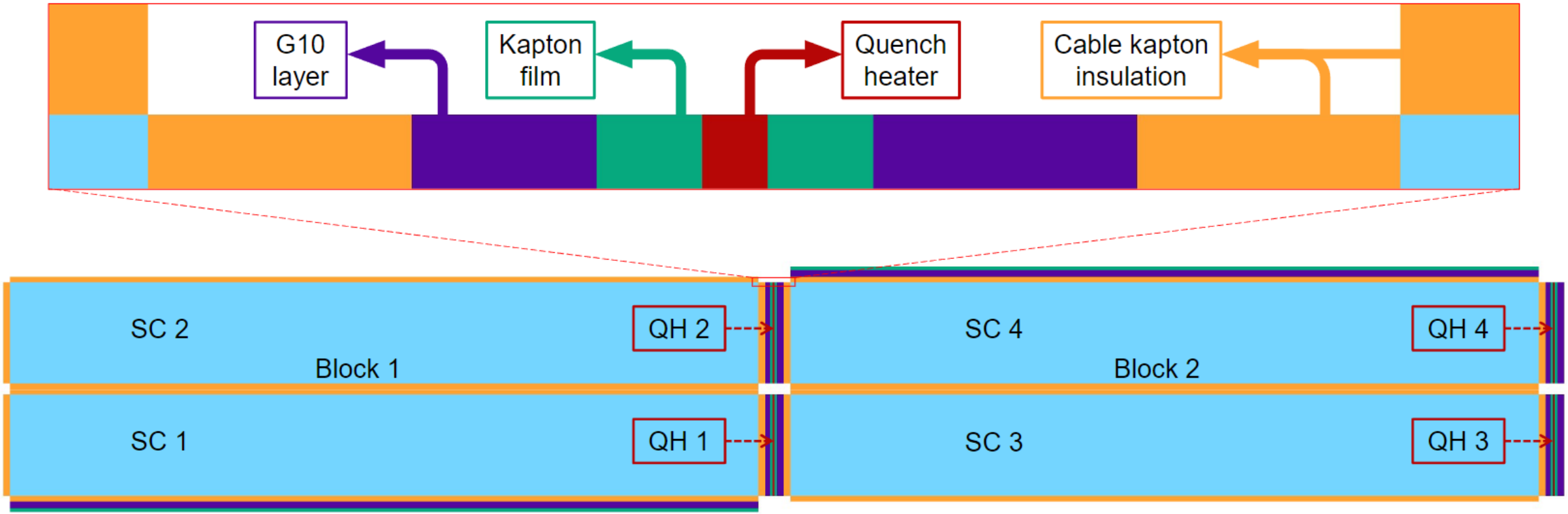
MBH

Thermal thin shell approximation (TSA)

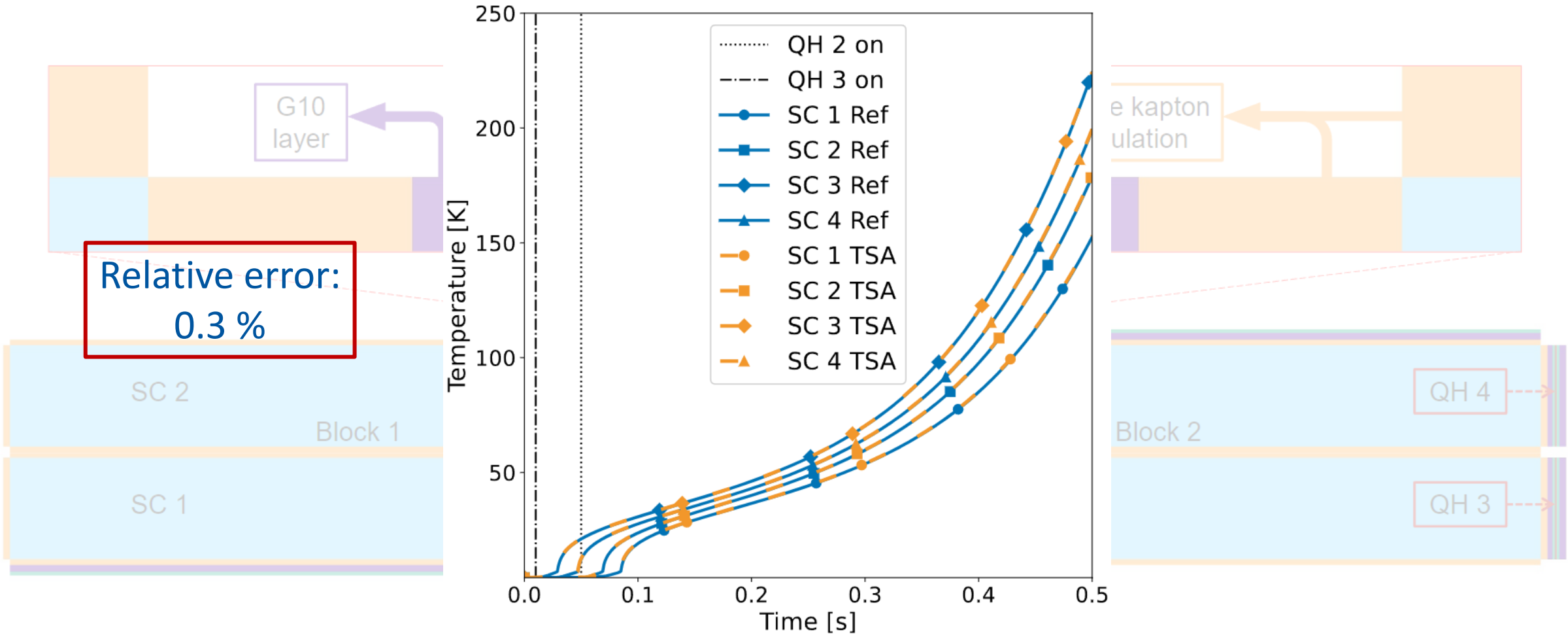
	Short-side cable insulation
	Long-side cable insulation
X	Fish bone
Y	Pole separator
	Cu spacer insulation
	Quench heater
	Quench heater films
Z	Ground insulation



TSA verification with 4-conductor model



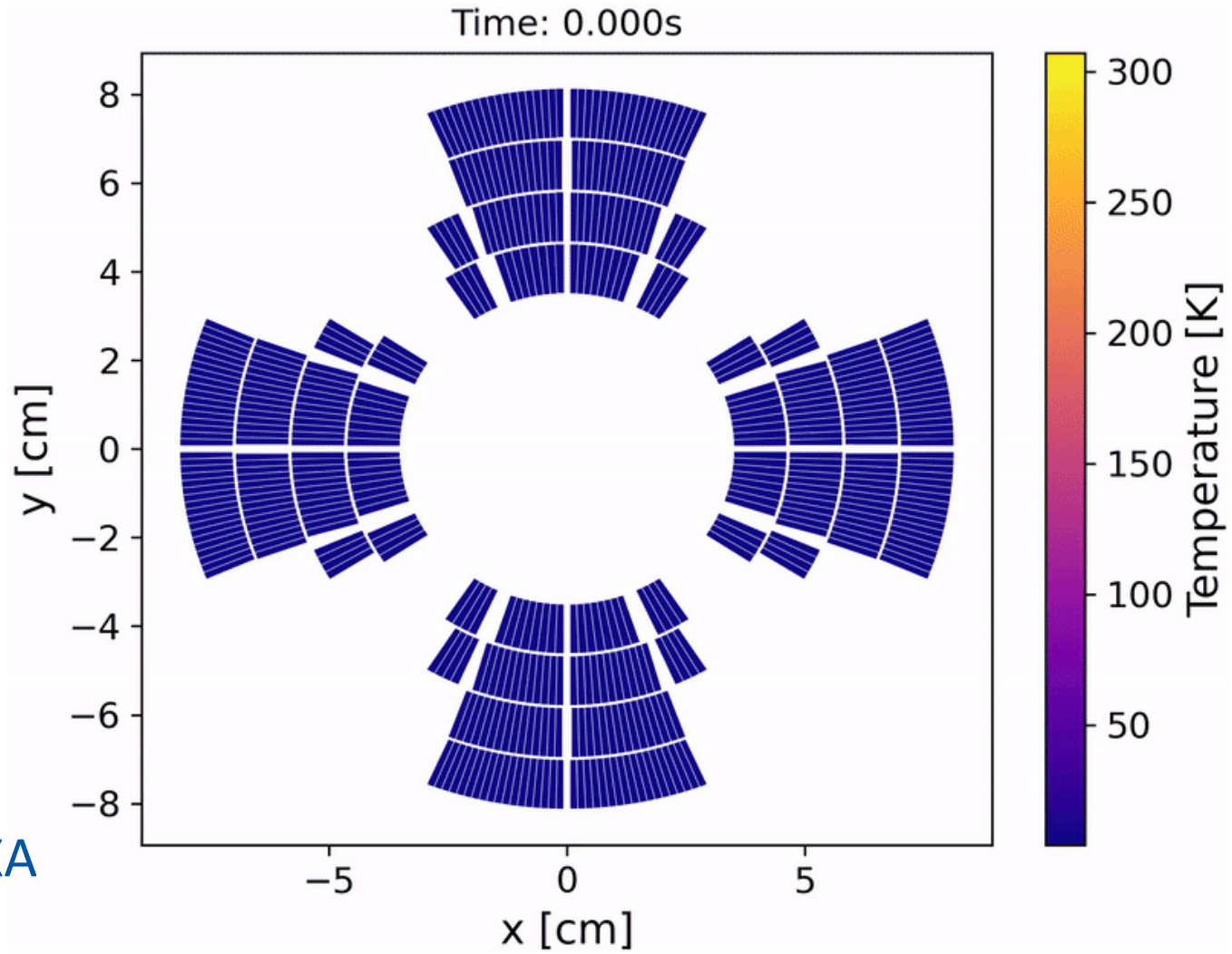
TSA verification - 4-conductor model



TSA verification - multi-pole

Overall relative error:
3 %

Relative error of hot-spot T:
1 %

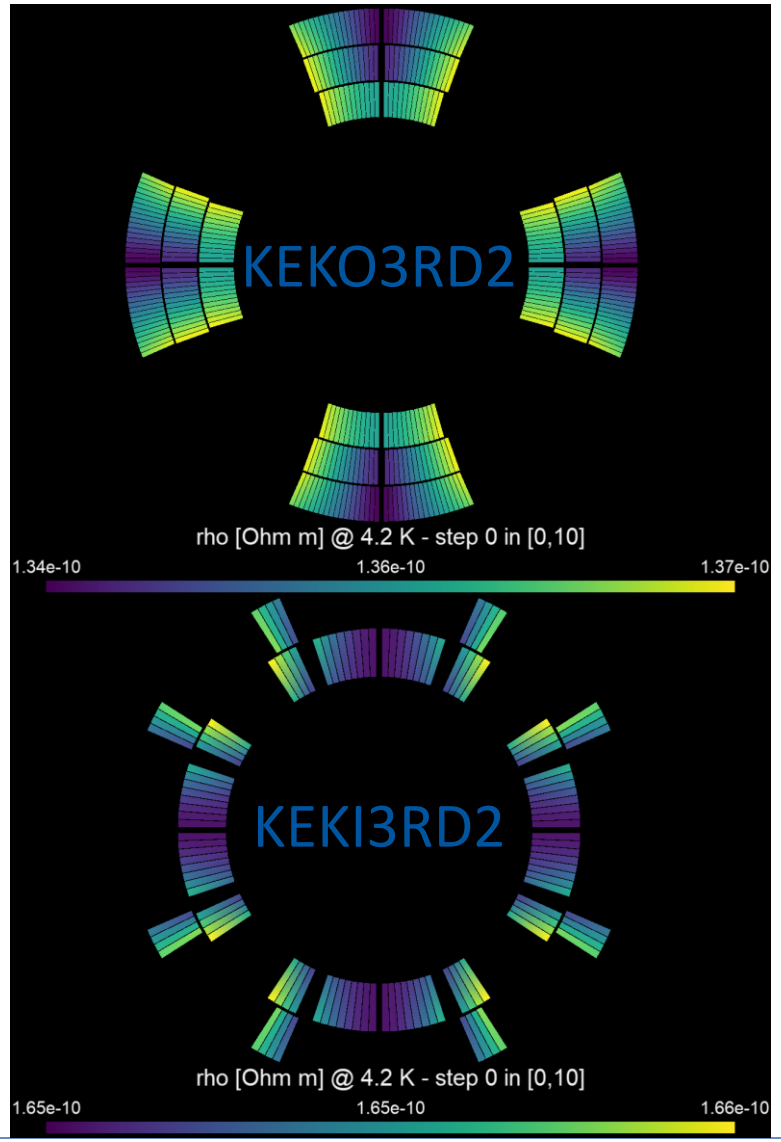


MQXA

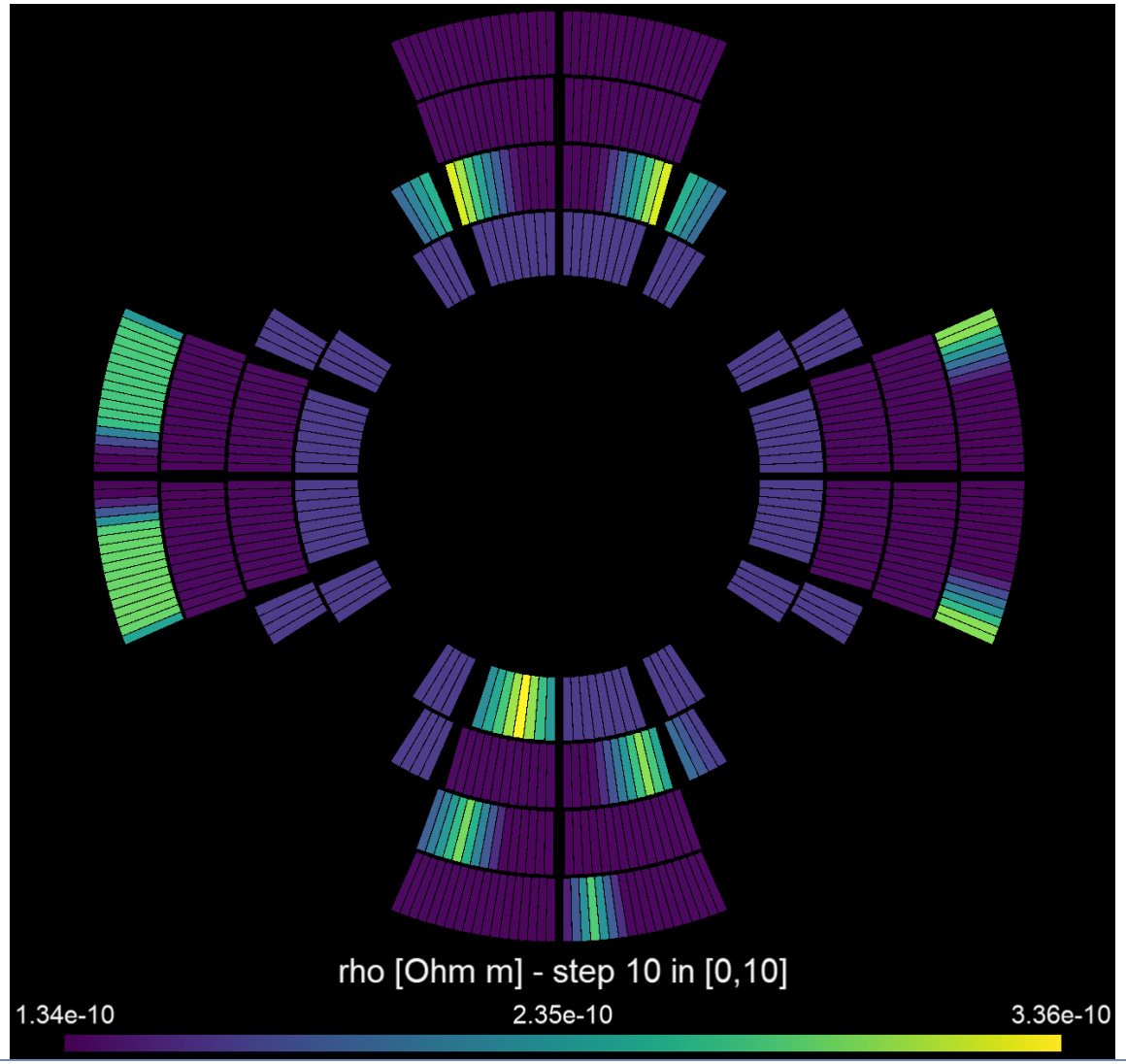
~40 times faster than meshed insulation model

Many thanks to Erik Schnaubelt

Multi-physics EM-TH simulations

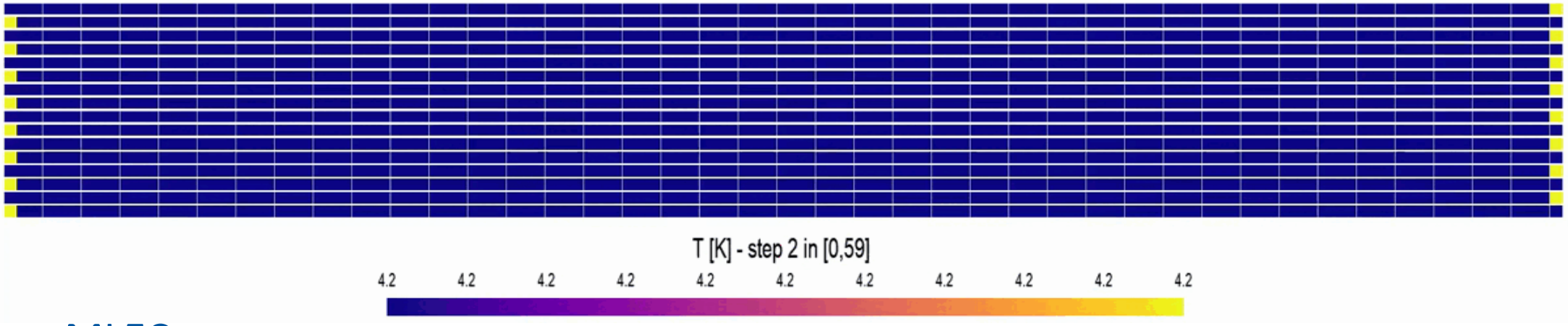


MQXA

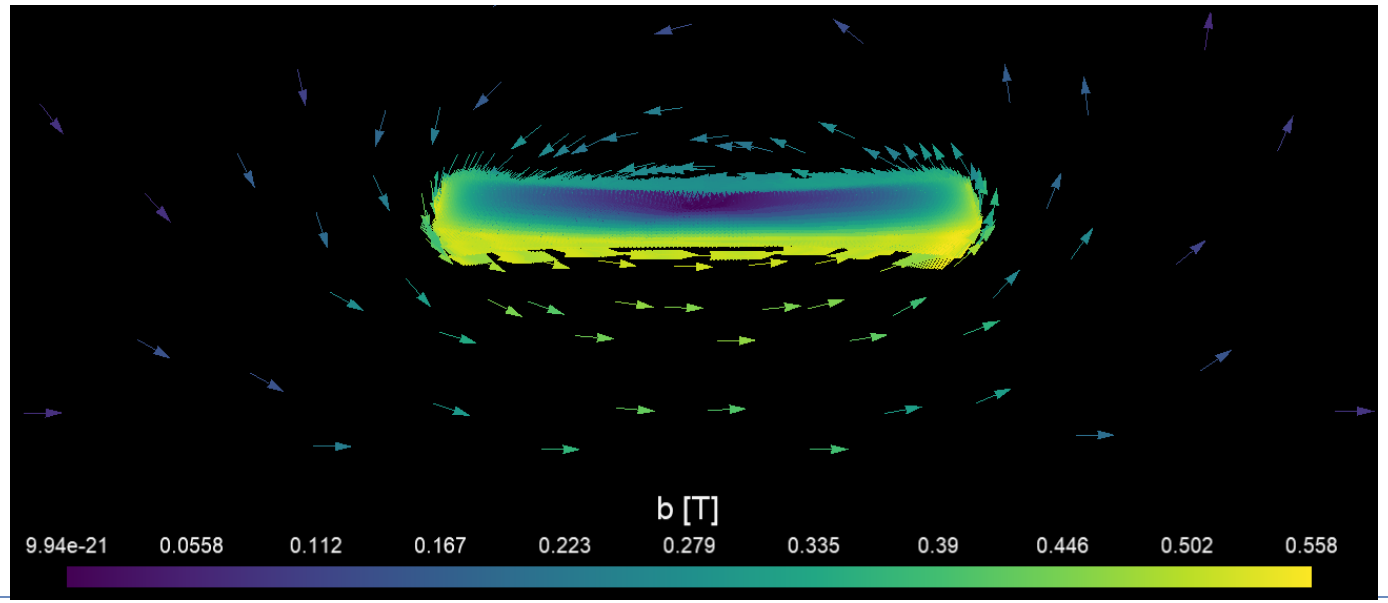


Many thanks to Erik Schnaubelt

FiQuS ~~multi-pole~~ Solenoid



MLEC



Conclusions

- Development support for the STEAM-SDK from the ground up
- Development support for FiQuS from the ground up
- Development of the complete CAE pipeline for multi-pole magnets
- Development support for the TSA model
- Plus, few honourable mentions (testing with unittest, reduced magnetic vector potential, symmetric models, …)

Many thanks to all former and current STEAM members!

special thanks to Mariusz Wozniak for his guidance and support