Brookhaven National Laboratory

U.S. DEPARTMENT of ENERGY

NSRL BO and Transitioning to A Digital Twin Implementation with Badger

Levente Hajdu*, R. Roussel

October 2025

Topics:

ADO’s / Multinet (the CAD control system)

Xopt and Badger

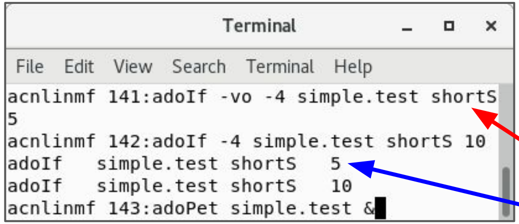
Badger interface with Multinet

Pitfalls of the prior BO \ Image processing approach

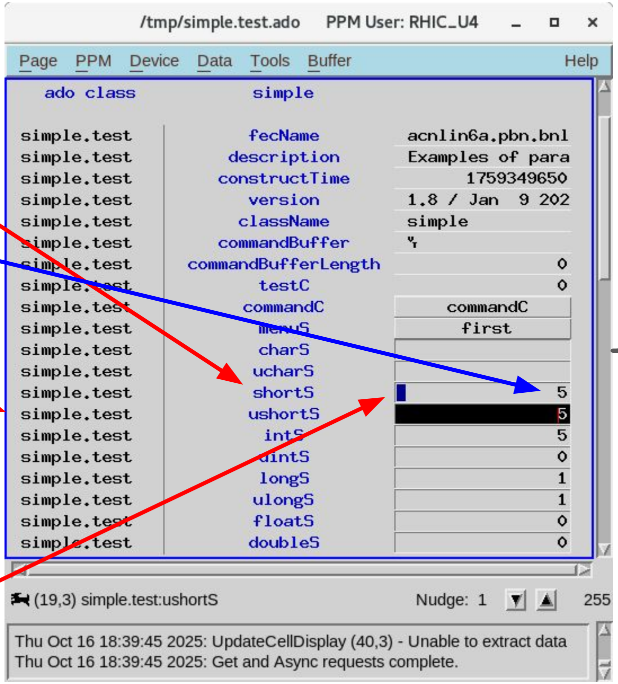
Accelerator Device Objects (ADO’s) and Multinet The Basics

- Multinet was selected in the 1990’s as the CAD home grown control system over EPICS for its implementation of multiplexing (PPM – Pulse to Pulse Modulated User)
  - Example: We can run NSRL, BLIP and fill RHIC in a multiplexed mode.
  - Packages for C\C++, Java, Python
    - Time based functions limited to C\C++ (we can bind to Python)
- EIC will use EPICS for it’s control system, but the injector chain will remain with ADO for some time until upgrade resource (funds / people / hardware) can he found
-

Accelerator Device Objects (ADO’s) and Multinet The Basics



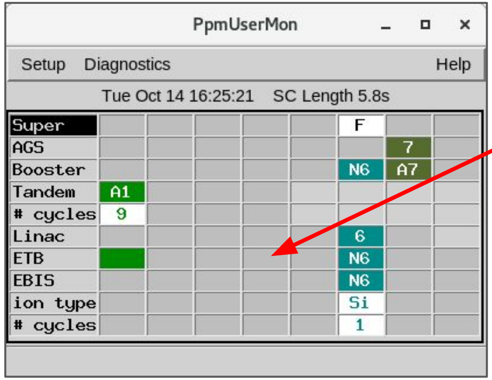
```
acnlinmf 141:adoIf -vo -4 simple.test shortS
5
acnlinmf 142:adoIf -4 simple.test shortS 10
adoIf    simple.test shortS    5
adoIf    simple.test shortS    10
acnlinmf 143:adoPet simple.test &
```


/tmp/simple.test.ado   PPM User: RHIC_U4
Page PPM Device Data Tools Buffer Help

ado class	simple	
simple.test	fecName	acnlin6a.pbn.bnl
simple.test	description	Examples of para
simple.test	constructTime	1759349650
simple.test	version	1.8 / Jan 9 202
simple.test	className	simple
simple.test	commandBuffer	Ÿ
simple.test	commandBufferLength	0
simple.test	testC	0
simple.test	commandC	commandC
simple.test	menuS	first
simple.test	charS	
simple.test	ucharS	
simple.test	shortS	5
simple.test	ushortS	5
simple.test	intS	5
simple.test	uintS	0
simple.test	longS	1
simple.test	ulongS	1
simple.test	floatS	0
simple.test	doubleS	0

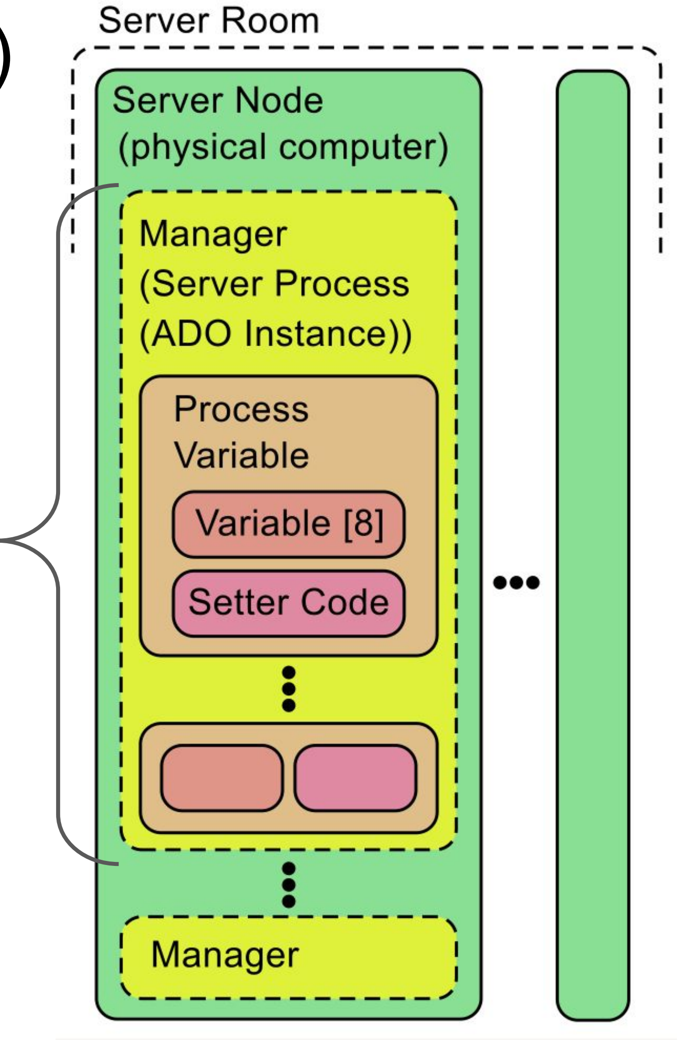
(19,3) simple.test:ushortS   Nudge: 1 ▼ ▲ 255

```
Thu Oct 16 18:39:45 2025: UpdateCellDisplay (40,3) - Unable to extract data
Thu Oct 16 18:39:45 2025: Get and Async requests complete.
```



Tue Oct 14 16:25:21   SC Length 5.8s

Super						F		
AGS							7	
Booster						N6	A7	
Tandem	A1							
# cycles	9							
Linac						6		
ETB	🟩					N6		
EBIS						N6		
ion type						Si		
# cycles						1		

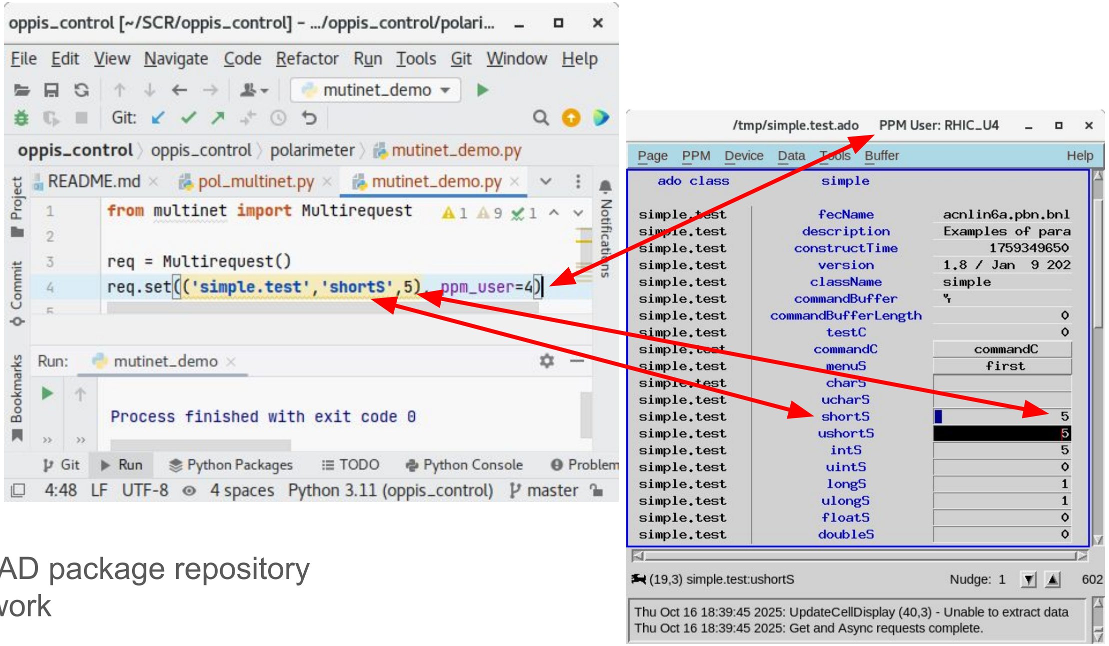


Accelerator Device Objects (ADO’s) and Multinet The Basics

- Multinet is the Python package for talking to ADO’s

```
from multinet import Multirequest

req = Multirequest()
req.set(('simple.test', 'shortS', 5), ppm_user=4)
```

Yes, this could easily be replaced with an epics function call!



oppis_control [~/SCR/oppis_control] – .../oppis_control/polari...

oppis_control 〉 oppis_control 〉 polarimeter 〉 mutinet_demo.py

```
from multinet import Multirequest

req = Multirequest()
req.set(('simple.test','shortS',5), ppm_user=4)
```

Run: mutinet_demo

```
Process finished with exit code 0
```

4:48 LF UTF-8 4 spaces Python 3.11 (oppis_control) master

/tmp/simple.test.ado PPM User: RHIC_U4

ado class	simple	
simple.test	fecName	acnlin6a.pbn.bnl
simple.test	description	Examples of para
simple.test	constructTime	1759349650
simple.test	version	1.8 / Jan 9 202
simple.test	className	simple
simple.test	commandBuffer	␀
simple.test	commandBufferLength	0
simple.test	testC	0
simple.test	commandC	commandC
simple.test	menuS	first
simple.test	charS	
simple.test	ucharS	
simple.test	shortS	5
simple.test	ushortS	5
simple.test	intS	5
simple.test	uintS	0
simple.test	longS	1
simple.test	ulongS	1
simple.test	floatS	0
simple.test	doubleS	0

(19,3) simple.test:ushortS Nudge: 1 602

```
Thu Oct 16 18:39:45 2025: UpdateCellDisplay (40,3) - Unable to extract data
Thu Oct 16 18:39:45 2025: Get and Async requests complete.
```

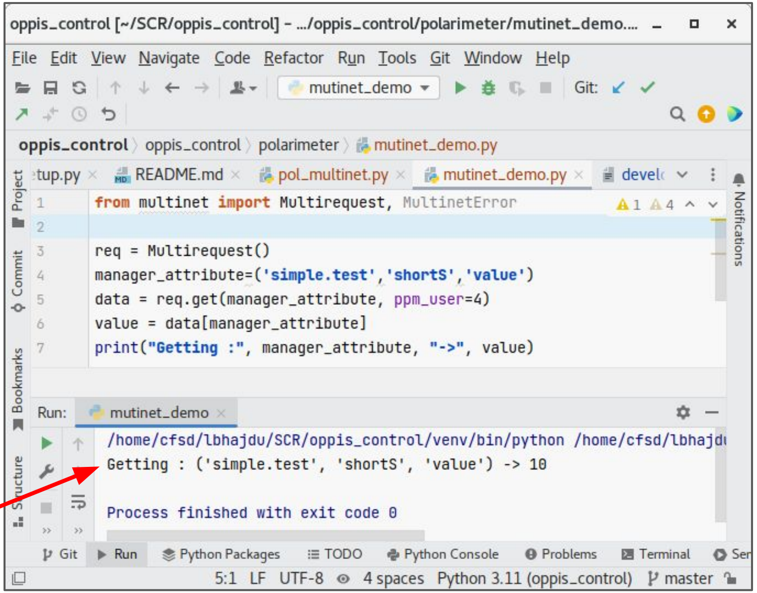
- This assumes the developer has access to the CAD package repository
- This assumes you’re on the internal controls network
- Error checking is omitted

Accelerator Device Objects (ADO's) and Multinet The Basics

- Multinet is the Python package for talking to ADO's

```python
from multinet import Multirequest
req = Multirequest()
manager_attribute=('simple.test', 'shortS', 'value')
data = req.get(manager_attribute, ppm_user=4)
value = data[manager_attribute]
print("Getting :", manager_attribute, "->", value)
```

We generally encapsulate these lines into a function


oppis_control [~/SCR/oppis_control] - .../oppis_control/polarimeter/mutinet_demo...
File Edit View Navigate Code Refactor Run Tools Git Window Help
mutinet_demo
Git:
oppis_control 〉 oppis_control 〉 polarimeter 〉 mutinet_demo.py
tup.py README.md pol_multinet.py mutinet_demo.py devel
Project
Commit
Bookmarks
Structure
Notifications
Run: mutinet_demo
Git Run Python Packages TODO Python Console Problems Terminal Ser
5:1 LF UTF-8 4 spaces Python 3.11 (oppis_control) master

```python
from multinet import Multirequest, MultinetError

req = Multirequest()
manager_attribute=('simple.test','shortS','value')
data = req.get(manager_attribute, ppm_user=4)
value = data[manager_attribute]
print("Getting :", manager_attribute, "->", value)
```

```
/home/cfsd/lbhajdu/SCR/oppis_control/venv/bin/python /home/cfsd/lbhajdu
Getting : ('simple.test', 'shortS', 'value') -> 10

Process finished with exit code 0
```

Output

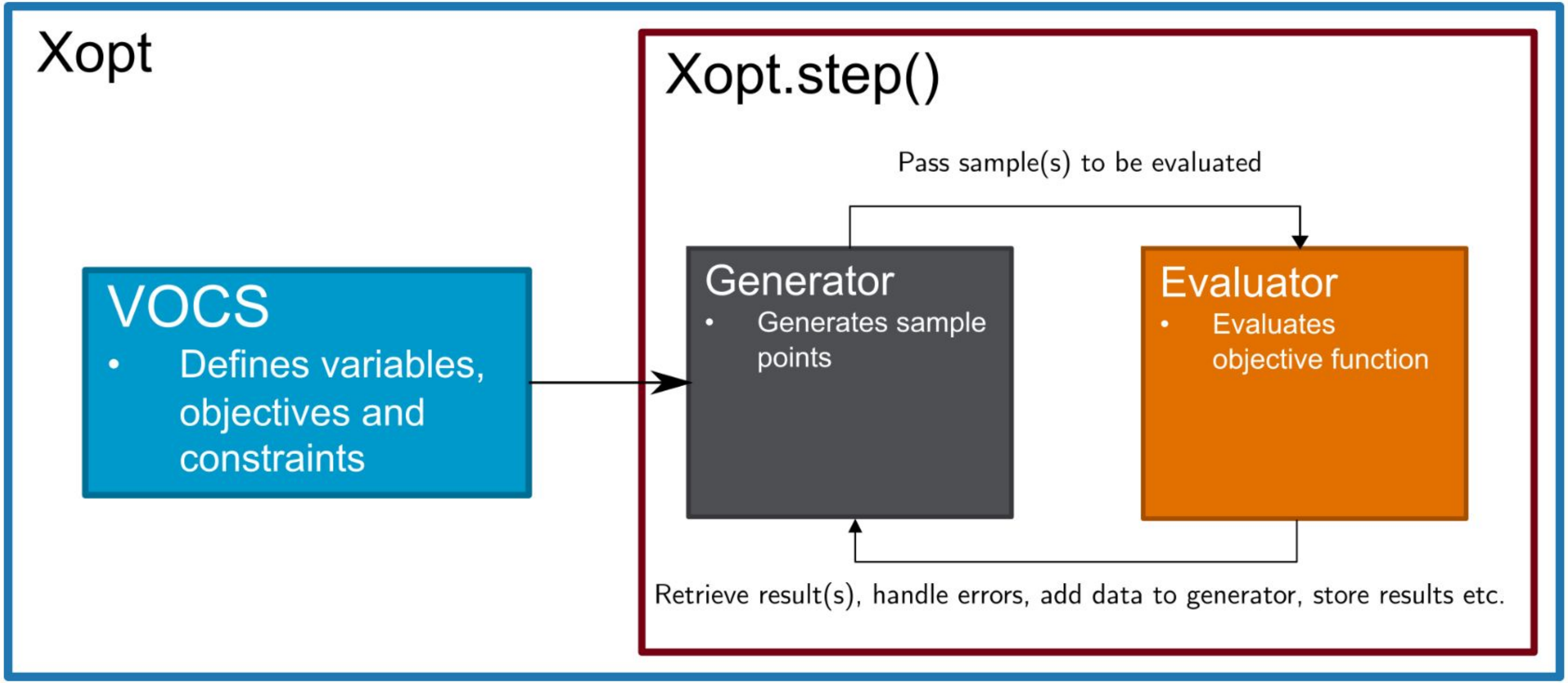
- This assumes the developer has access to the CAD package repository
- This assumes you're on the internal controls network
- Error checking is omitted

What is Xopt?

- Flexible, open-source **framework** for optimization of arbitrary problems using python
- **Independent** of problem type (simulation or experiment)
- **Independent** of optimization algorithm + easy to incorporate custom algorithms
- **Easy to use** text interface and/or advanced customized use for professionals

https://github.com/xopt-org/Xopt

Xopt Overview - Structure


Xopt
Xopt.step()
Pass sample(s) to be evaluated
VOCS
• Defines variables, objectives and constraints
Generator
• Generates sample points
Evaluator
• Evaluates objective function
Retrieve result(s), handle errors, add data to generator, store results etc.

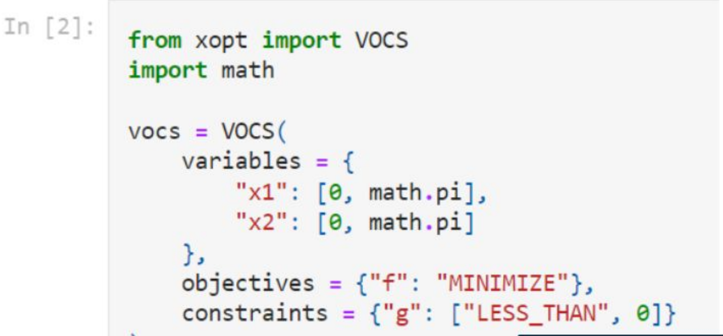
Note: this process can also be done asynchronously

Xopt Overview – Example problem

Define the domain/goals

$$x_1, x_2 \in [0, \pi] \qquad \boldsymbol{x}^* = arg\,min\, f(\boldsymbol{x})$$

$$g(\boldsymbol{x}) \leq 0$$



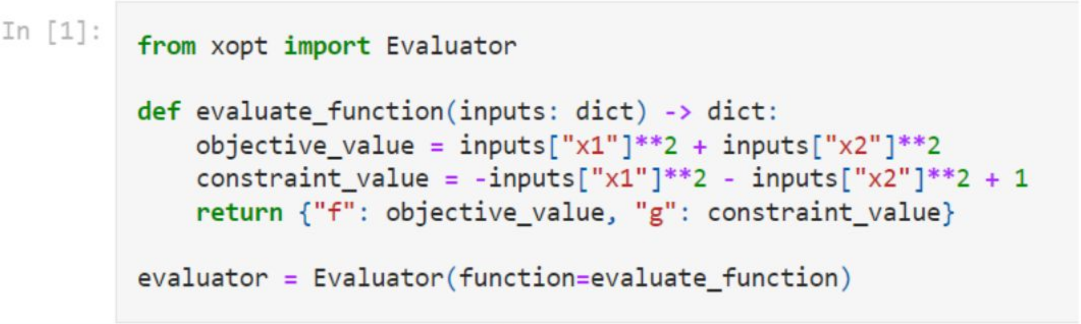
In [2]:

```
from xopt import VOCS
import math

vocs = VOCS(
    variables = {
        "x1": [0, math.pi],
        "x2": [0, math.pi]
    },
    objectives = {"f": "MINIMIZE"},
    constraints = {"g": ["LESS_THAN", 0]}
)
```

Define the objectives/constraints

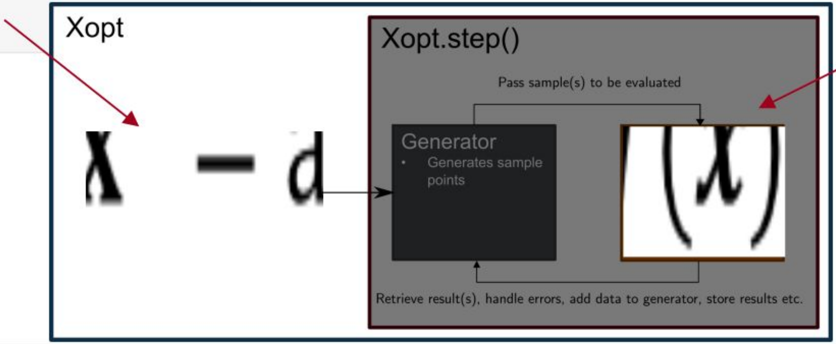
$$f(x_1, x_2) = x_1^2 + x_2^2$$

$$g(x_1, x_2) = 1 - x_1^2 - x_2^2$$



In [1]:

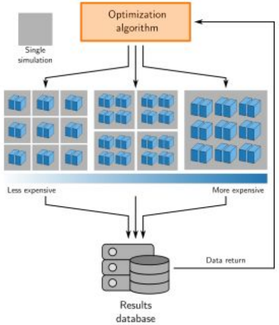
```
from xopt import Evaluator

def evaluate_function(inputs: dict) -> dict:
    objective_value = inputs["x1"]**2 + inputs["x2"]**2
    constraint_value = -inputs["x1"]**2 - inputs["x2"]**2 + 1
    return {"f": objective_value, "g": constraint_value}

evaluator = Evaluator(function=evaluate_function)
```


Xopt
Xopt.step()
Pass sample(s) to be evaluated
Generator
Generates sample points
Retrieve result(s), handle errors, add data to generator, store results etc.

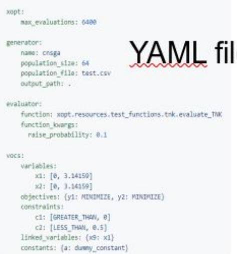
The Xopt Ecosystem

Xopt algorithm implementation

https://github.com/xopt-org/Xopt



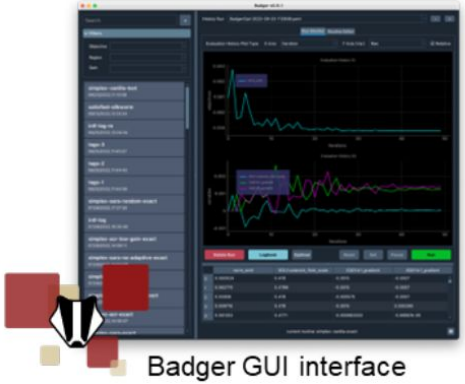
Accelerator simulation



```
xopt:
    max_evaluations: 6400

generator:
    name: cnsga
    population_size: 64
    population_file: test.csv
    output_path: .

evaluator:
    function: xopt.resources.test_functions.tnk.evaluate_TNK
    function_kwargs:
      raise_probability: 0.1

vocs:
    variables:
        x1: [0, 3.14159]
        x2: [0, 3.14159]
    objectives: {y1: MINIMIZE, y2: MINIMIZE}
    constraints:
        c1: [GREATER_THAN, 0]
        c2: [LESS_THAN, 0.5]
    linked_variables: {x9: x1}
    constants: {a: dummy_constant}
```

YAML file

Production ready control

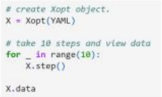


Badger GUI interface

https://github.com/xopt-org/Badger

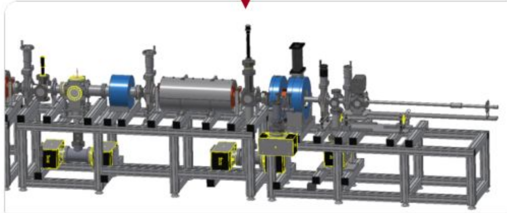
Online Control R&D

Python interface



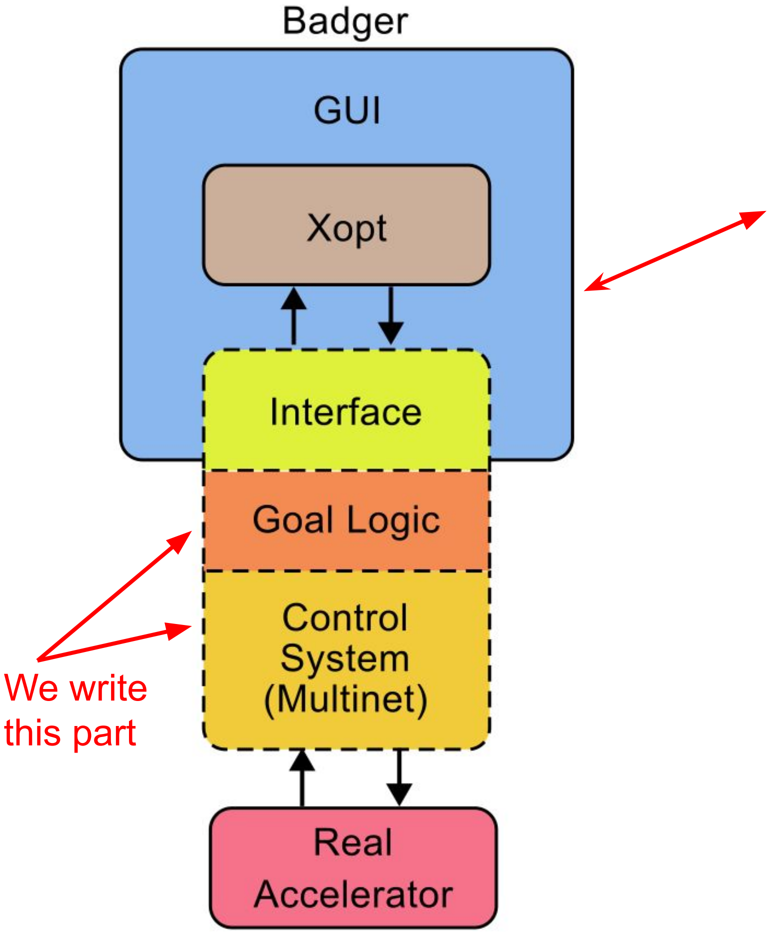
```
# create Xopt object.
X = Xopt(YAML)

# take 10 steps and view data
for _ in range(10):
    X.step()

X.data
```

Arbitrary problem



Experiment facility

The Badger - Multinet Interface



Badger

GUI

Xopt

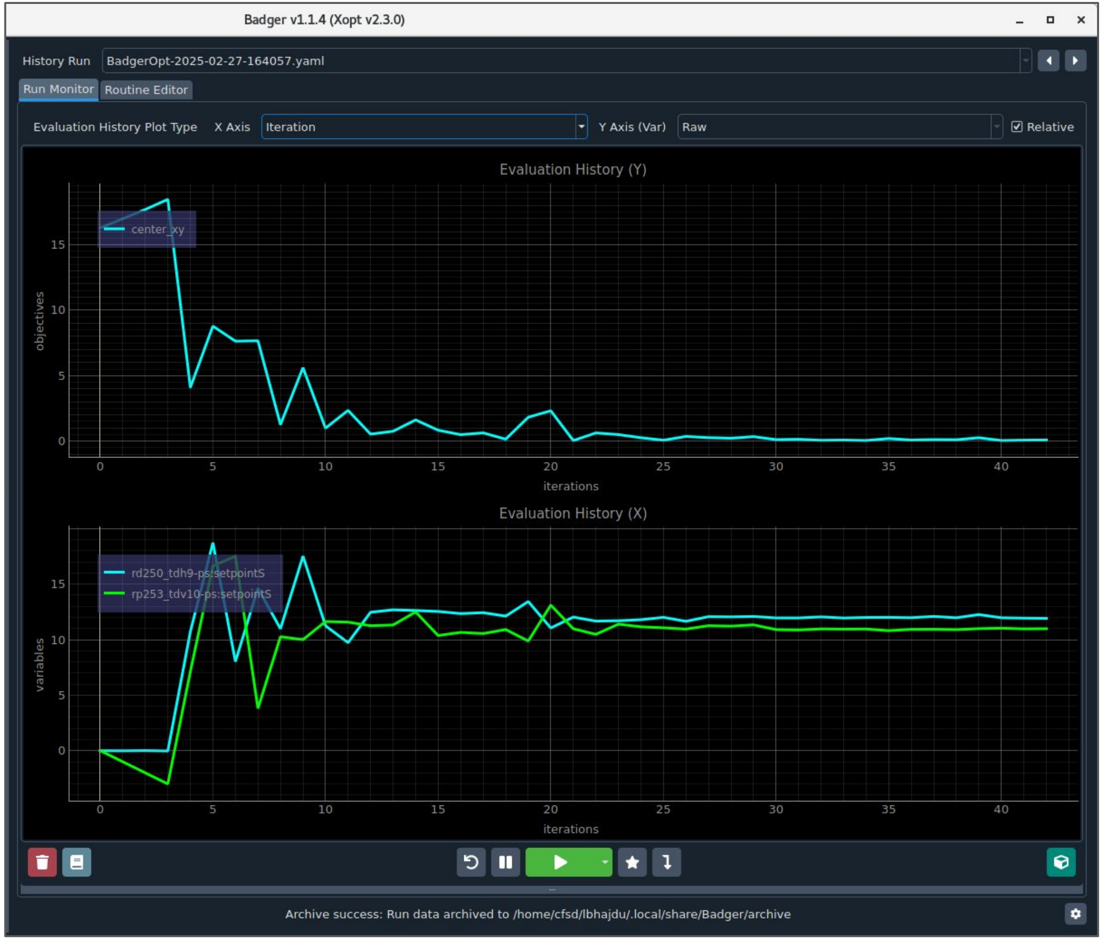
Interface

Goal Logic

Control System (Multinet)

Real Accelerator

We write this part



Badger v1.1.4 (Xopt v2.3.0)

History Run: BadgerOpt-2025-02-27-164057.yaml

Run Monitor | Routine Editor

Evaluation History Plot Type — X Axis: Iteration — Y Axis (Var): Raw — ☑ Relative

Archive success: Run data archived to /home/cfsd/lbhajdu/.local/share/Badger/archive

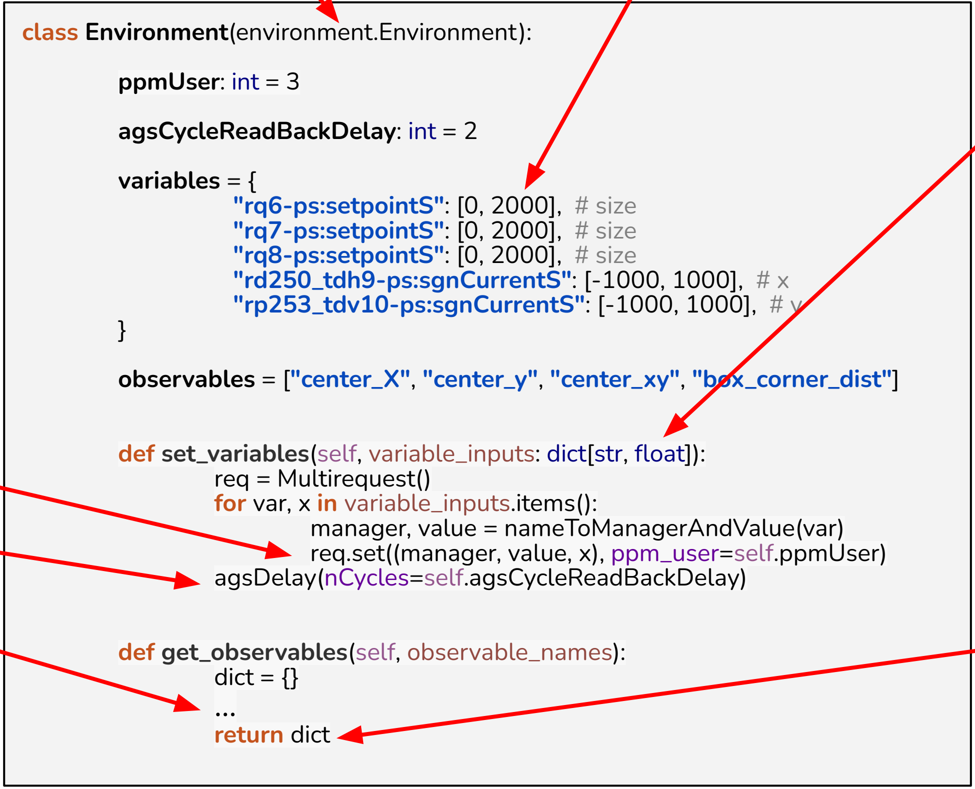
Interfacing Badger to Multinet

Base class to implement Badger interface

[min, max] we overwrite these

Badger passes us values to set as a dictionary:

```
"rd250_tdh9-ps:sgnCurrentS": 15,
"rp253_tdv10-ps:sgnCurrentS": 17
```



```
class Environment(environment.Environment):

        ppmUser: int = 3

        agsCycleReadBackDelay: int = 2

        variables = {
                "rq6-ps:setpointS": [0, 2000],  # size
                "rq7-ps:setpointS": [0, 2000],  # size
                "rq8-ps:setpointS": [0, 2000],  # size
                "rd250_tdh9-ps:sgnCurrentS": [-1000, 1000],  # x
                "rp253_tdv10-ps:sgnCurrentS": [-1000, 1000],  # y
        }

        observables = ["center_X", "center_y", "center_xy", "box_corner_dist"]


        def set_variables(self, variable_inputs: dict[str, float]):
                req = Multirequest()
                for var, x in variable_inputs.items():
                        manager, value = nameToManagerAndValue(var)
                        req.set((manager, value, x), ppm_user=self.ppmUser)
                agsDelay(nCycles=self.agsCycleReadBackDelay)


        def get_observables(self, observable_names):
                dict = {}
                ...
                return dict
```

Sets multinet values

Delay

This is more complex,
But it could just be
return a bpm value

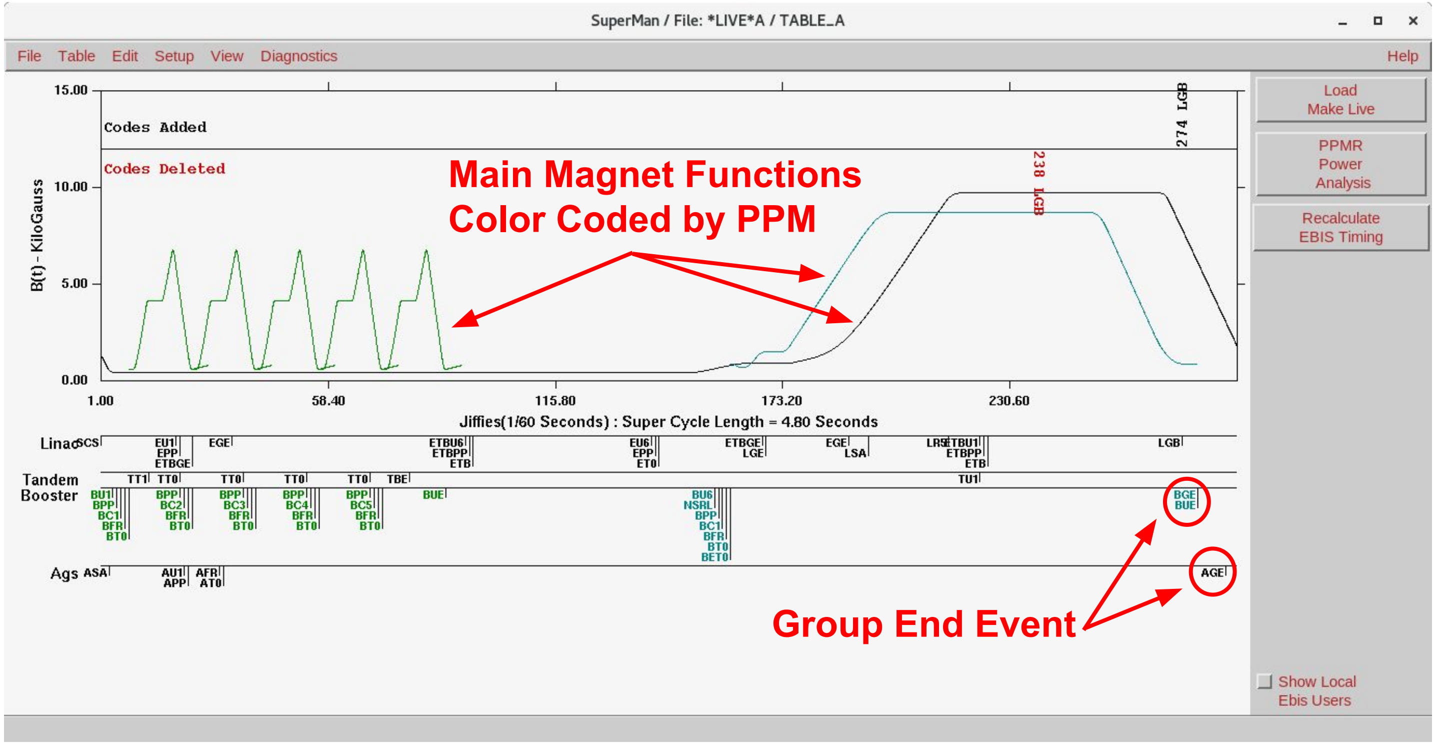
We pass back a dictionary of observables:

```
"center_X": 15,
"center_y": 17,
"center_XY": 25,
"box_corner_dist": 30
```

About the delay … it’s a synchrotron injector

Because we don’t want waveforms jumping suddenly updates are only applied on the very next cycle if they are set before the “group-end” event and then one cycle needs to happen before readback of the new data (ie 2 cycles min.)

Super Cycle Manager



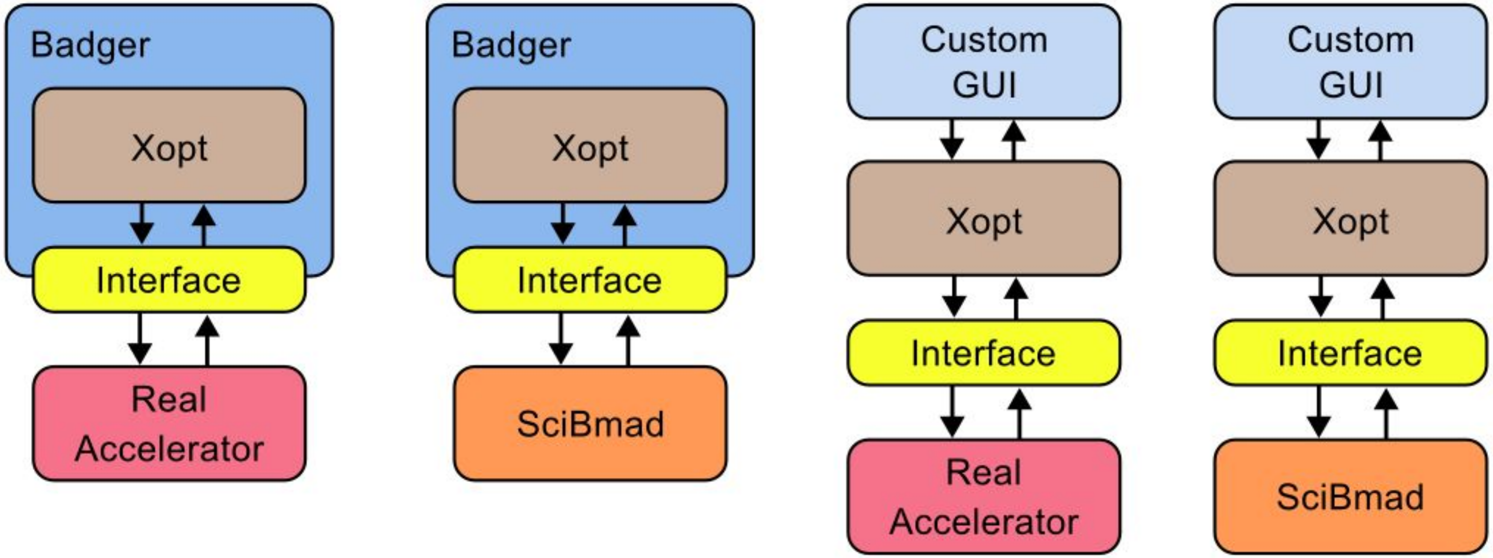
Main Magnet Functions Color Coded by PPM

Group End Event

Permutations

Writing the interface to “glue” parts together is not hard so many permutations are possible:

*Assuming knowledge of your control system



Badger
Xopt
Interface
Real Accelerator

Badger
Xopt
Interface
SciBmad

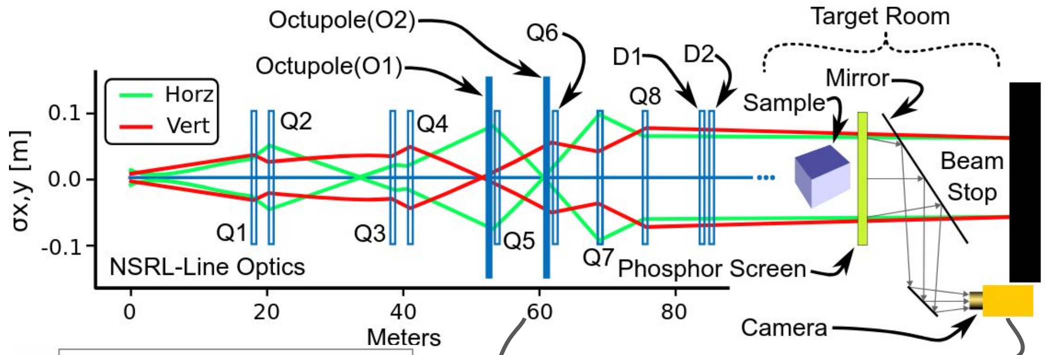
Custom GUI
Xopt
Interface
Real Accelerator

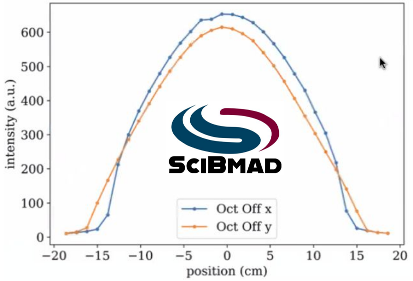
Custom GUI
Xopt
Interface
SciBmad

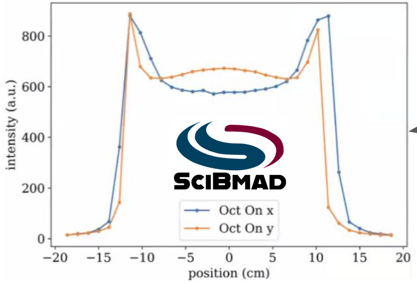
The Pitfalls of BO with Complex Image Processing

- Typically all problems arise from reporting the wrong number back to Badger
- If we had it to do all over again we could get closer
  - but in some cases it would fail to converge
- A digital twin we can adjust offline will save beam time
- We may still need image processing with BO for the final step where there is tolerance drift over time
  - SciBmad will not model the drift over time, unless the model is adjusted

NSRL - Beamline Optics







Bmad simulation of beam profile for 211 MeV/u bismuth with octupoles on and off.

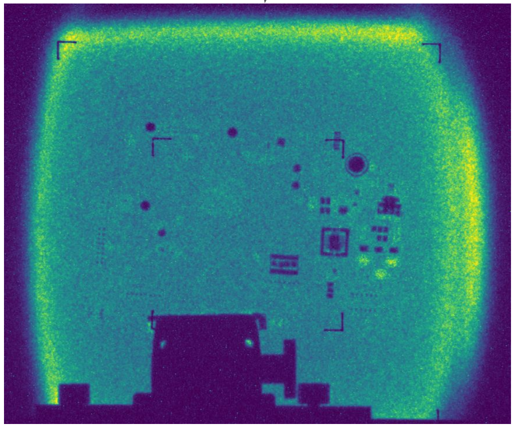
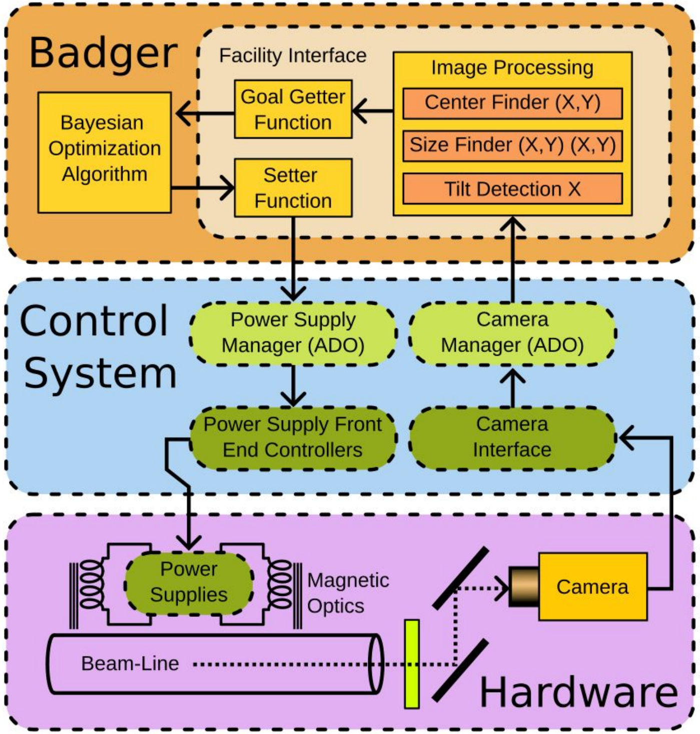


Image taken with NSRL beam imager showing PCB board being tested while being bombarded with 385MeV/u bismuth ions.

Data flow between Badger, Control System, and Hardware



Badger

Facility Interface

Bayesian Optimization Algorithm

Goal Getter Function

Setter Function

Image Processing

Center Finder (X,Y)

Size Finder (X,Y) (X,Y)

Tilt Detection X

Control System

Power Supply Manager (ADO)

Camera Manager (ADO)

Power Supply Front End Controllers

Camera Interface

Power Supplies

Magnetic Optics

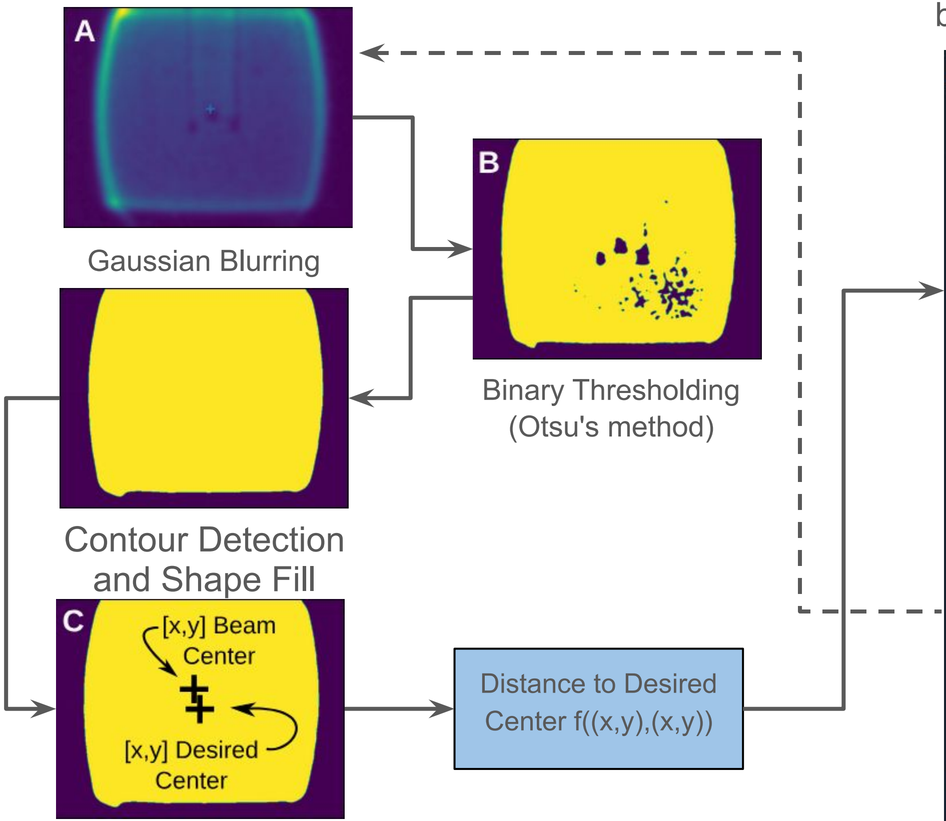
Beam-Line

Camera

Hardware

Beam species, ranging in atomic number (Z) from 1, hydrogen/protons, to 83 bismuth beams up to 20cm by 20cm uniform-area

Beam Centering Goal Algorithm



Gaussian Blurring

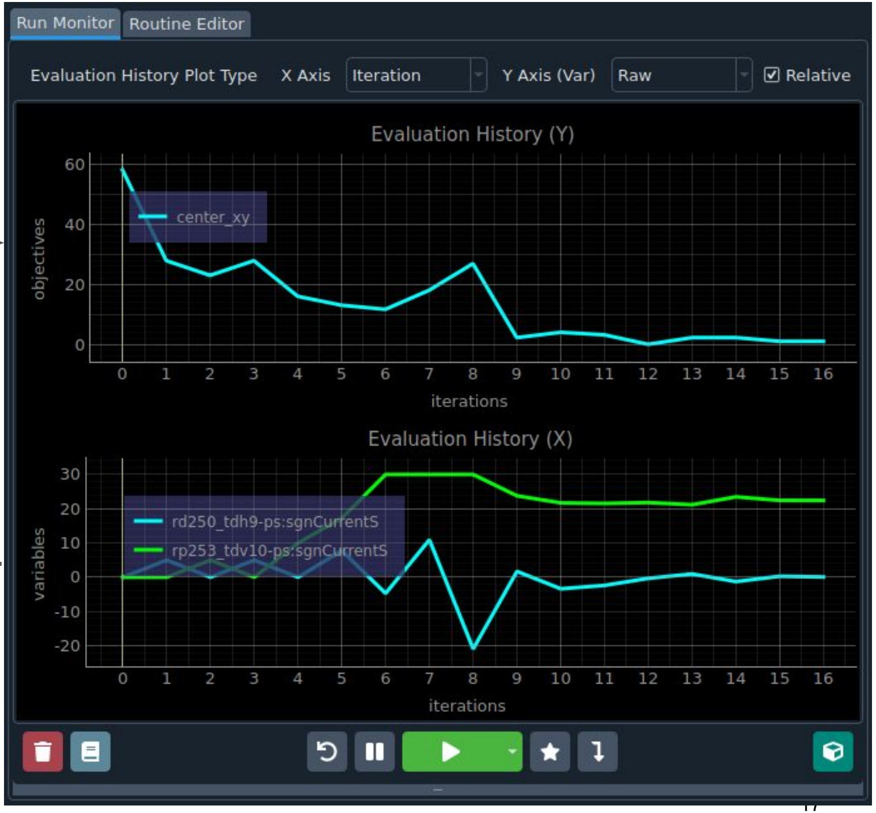
Binary Thresholding (Otsu's method)

Contour Detection and Shape Fill

Centroid Calculation of X and Y Moments

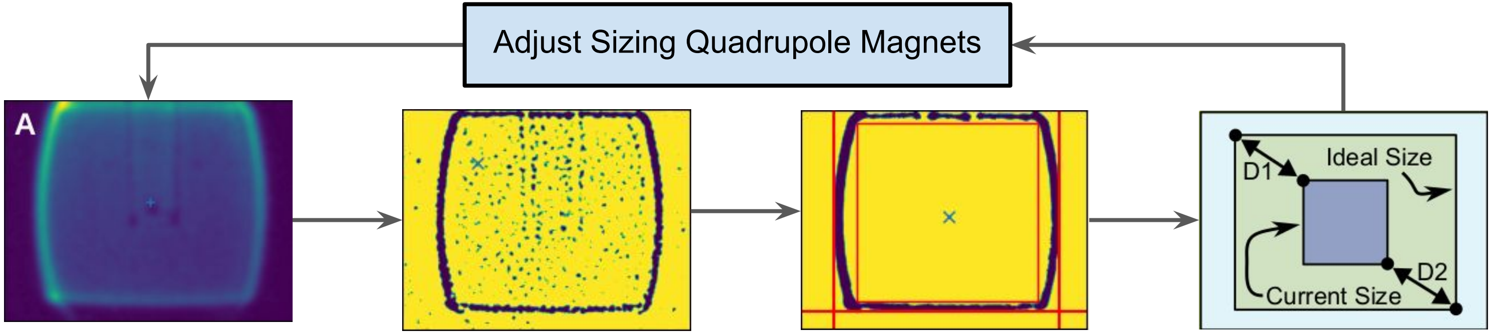
Distance to Desired Center f((x,y),(x,y))

Badger interface showing a beam centering study: the top plot displays the beam’s distance from the center, while the bottom plot shows dipole settings.



Evaluation History Plot Type X Axis Iteration Y Axis (Var) Raw ☑ Relative

Usable Area Goal Quantification Algorithm

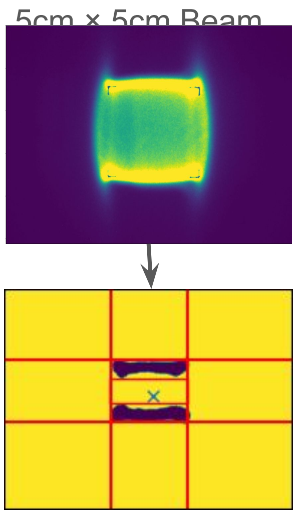


Heavy Gaussian Blur

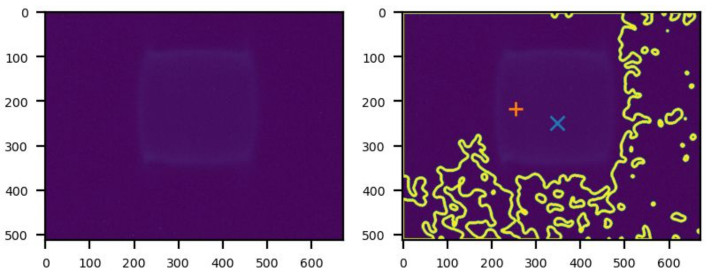
Adaptive Threshold (Gaussian-Weighted Neighbours)

1)Remove Pixel Islands
2) Scanning Four Outer Sides
3) Grow Out Box from Center
(Found in Prior Center Finder Step)

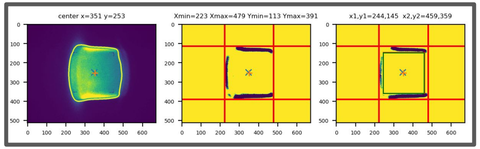
Calculate Distance to Corners of Ideal Size



It still works with only 2 sides found



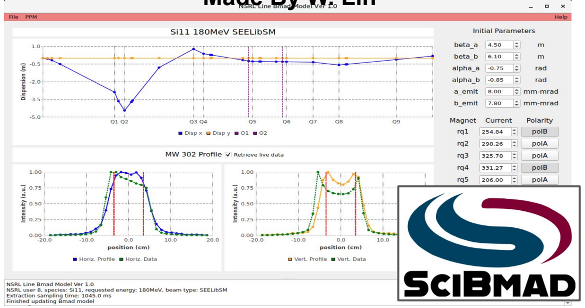
Signal Too Small (Will Not Converge)



Typical debugging strip printed out with each spill

Future Work: A True Digital Twin

Made By W. Lin



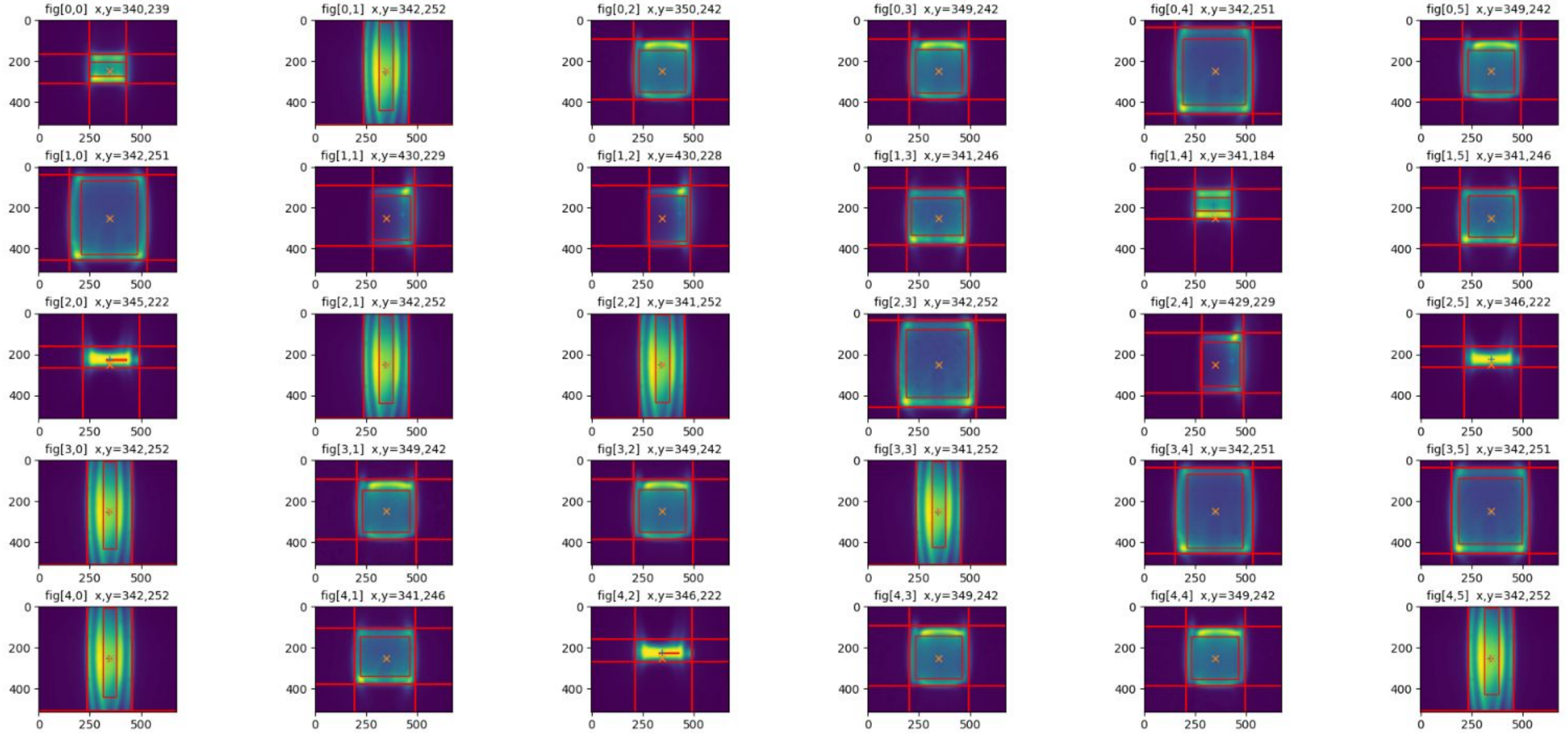
Initial Parameters

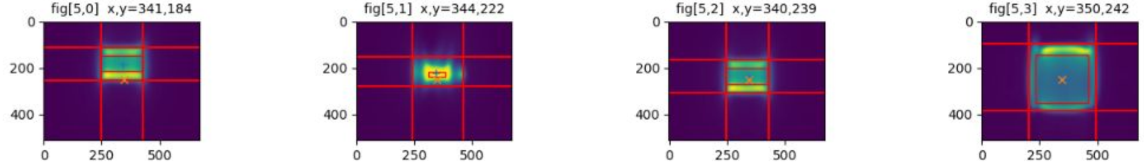
beta_a	4.50	m
beta_b	6.10	m
alpha_a	-0.75	rad
alpha_b	-0.85	rad
a_emit	8.00	mm-mrad
b_emit	7.80	mm-mrad

Magnet	Current	Polarity
rq1	254.84	polB
rq2	298.26	polA
rq3	325.78	polA
rq4	331.27	polB
rq5	206.00	polA

```
NSRL Line Bmad Model Ver 1.0
NSRL user 8, species: Si11, requested energy: 180MeV, beam type: SEELibSM
Extraction sampling time: 1045.0 ms
Finished updating Bmad model
```

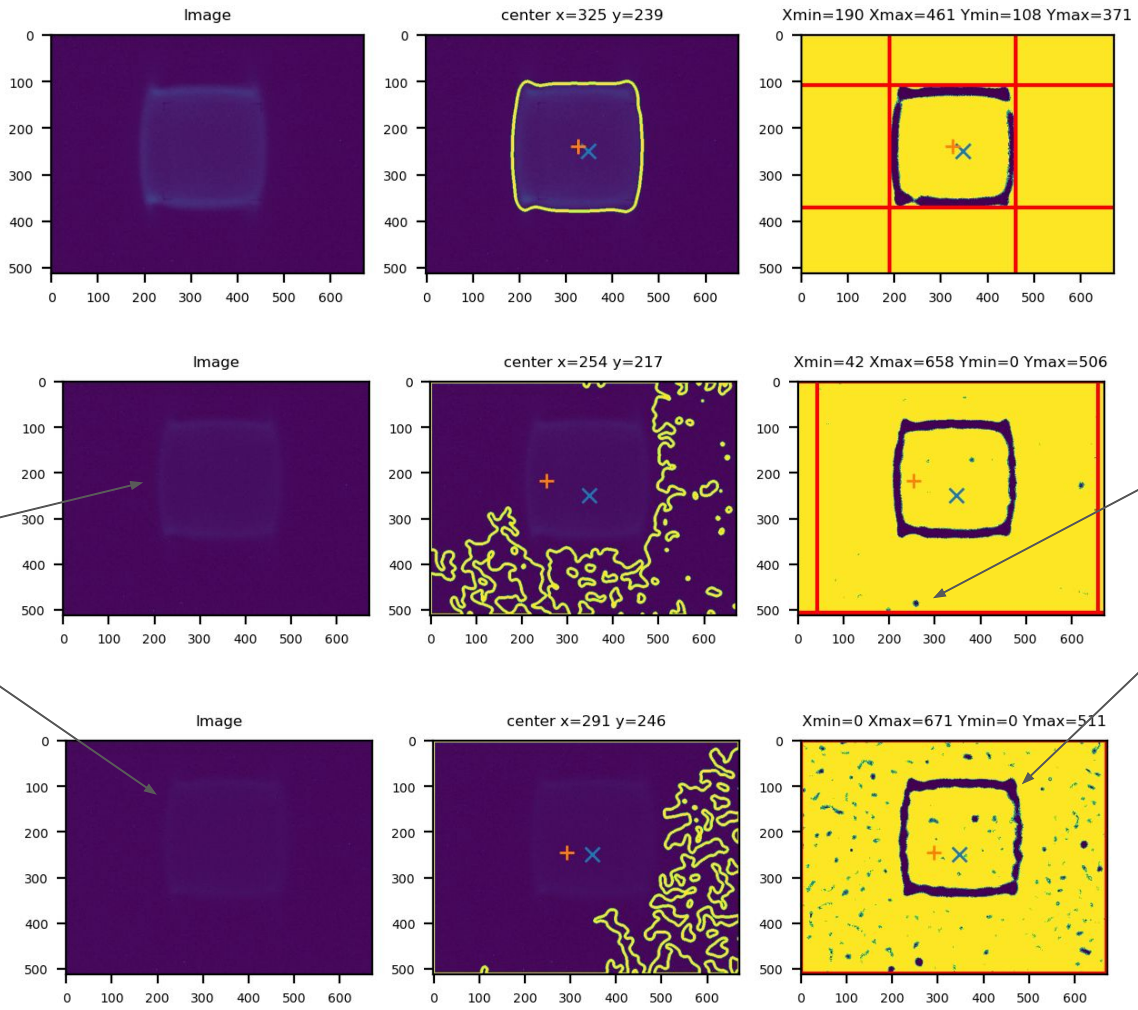
SciBMAD





We thought we had good offline test until we tied it live (species, aperture,energy)

Camera Aperture Stop-Down Test

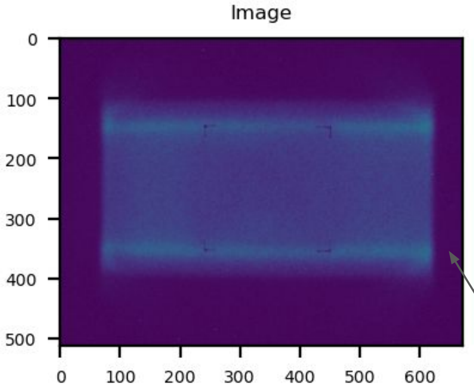


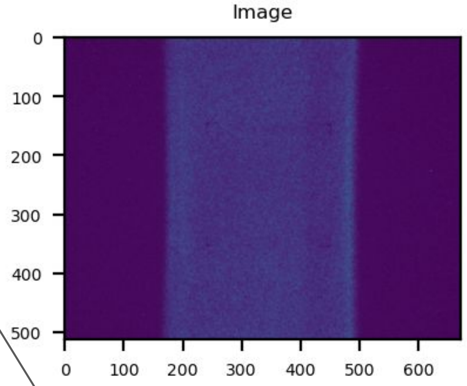
The human eye can still see the shape

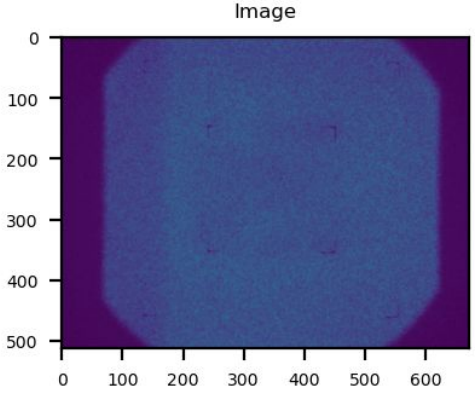
This is when we released just finding the right threshold was not going to cut it and we need to remove small islands of pixels.

If we had it to do over we would have used the Gaussian-Weighted neighbours threshold over Otsu's method for thresholding

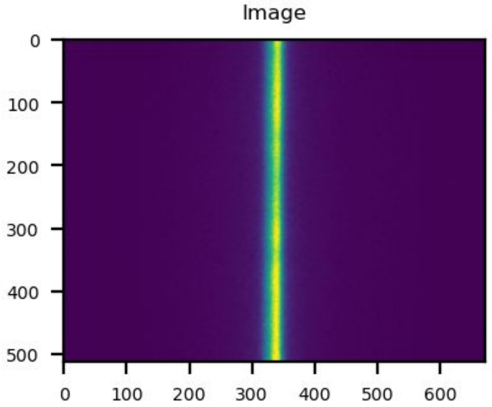
Too Much Travel



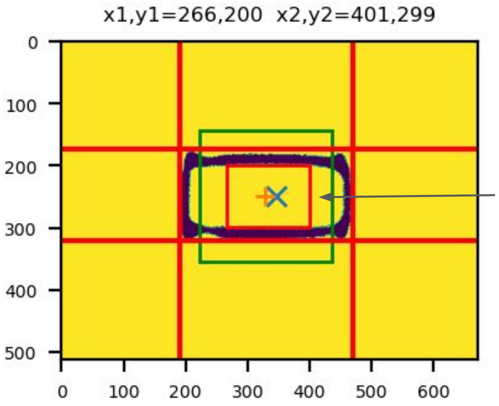




The whole beam pipe

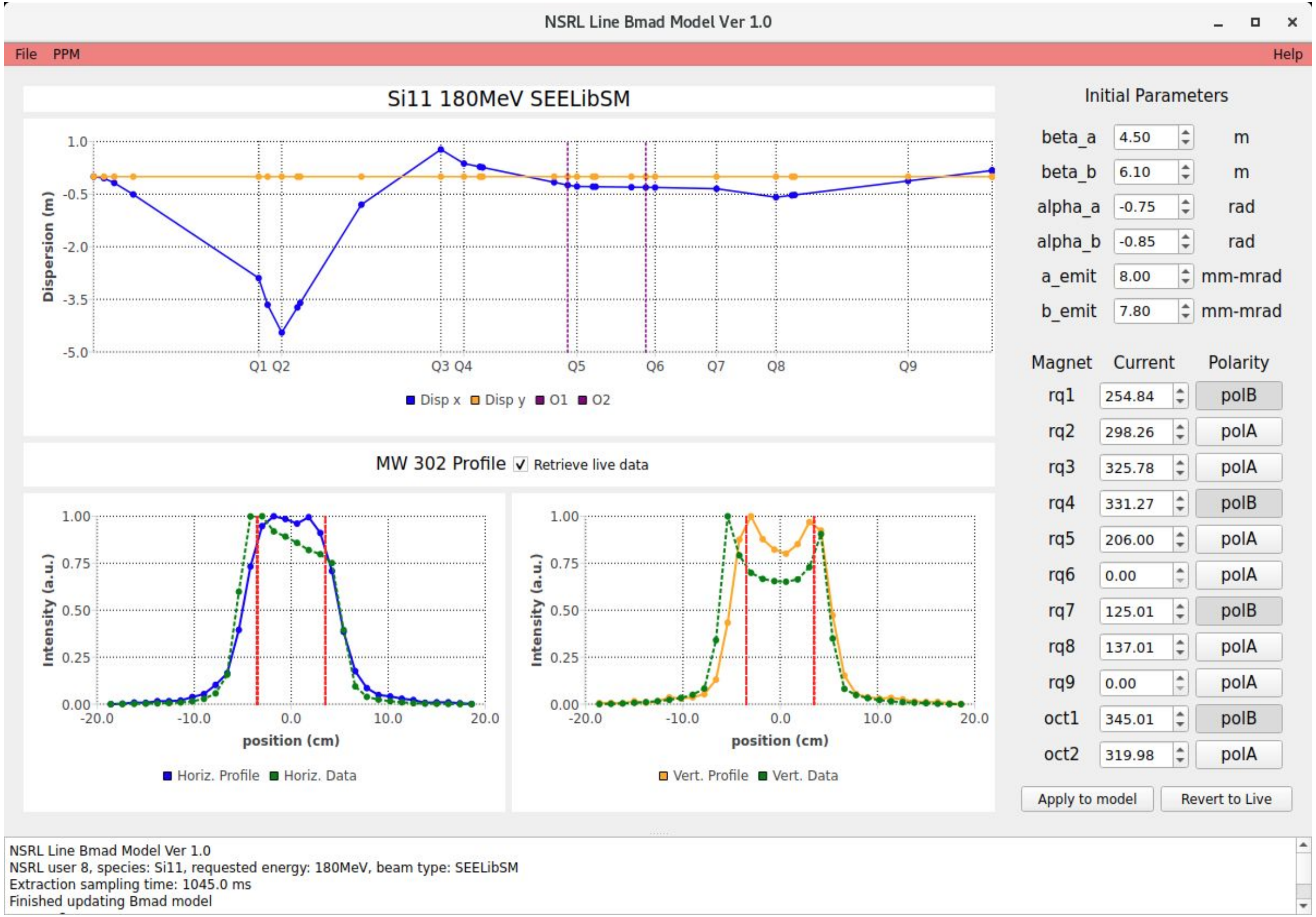


Object it the way, we could have moved this



Wrong box expansion

The Next Step … A True Digital Twin … By Lucy



☑ Retrieve live data

Initial Parameters

beta_a	4.50	m
beta_b	6.10	m
alpha_a	-0.75	rad
alpha_b	-0.85	rad
a_emit	8.00	mm-mrad
b_emit	7.80	mm-mrad

Magnet	Current	Polarity
rq1	254.84	polB
rq2	298.26	polA
rq3	325.78	polA
rq4	331.27	polB
rq5	206.00	polA
rq6	0.00	polA
rq7	125.01	polB
rq8	137.01	polA
rq9	0.00	polA
oct1	345.01	polB
oct2	319.98	polA

```
NSRL Line Bmad Model Ver 1.0
NSRL user 8, species: Si11, requested energy: 180MeV, beam type: SEELibSM
Extraction sampling time: 1045.0 ms
Finished updating Bmad model
```