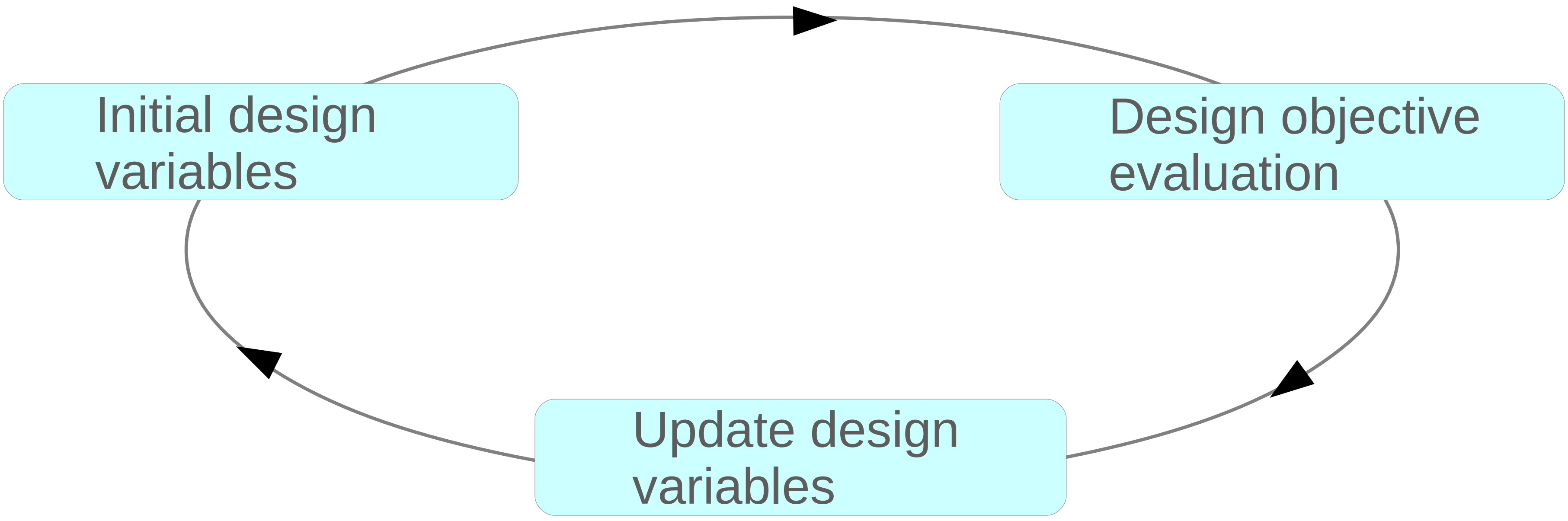
Muon Injection Optimzation in Simulations

Ritwika Chakraborty (PSI)

MuEDM collaboration Spring Meeting

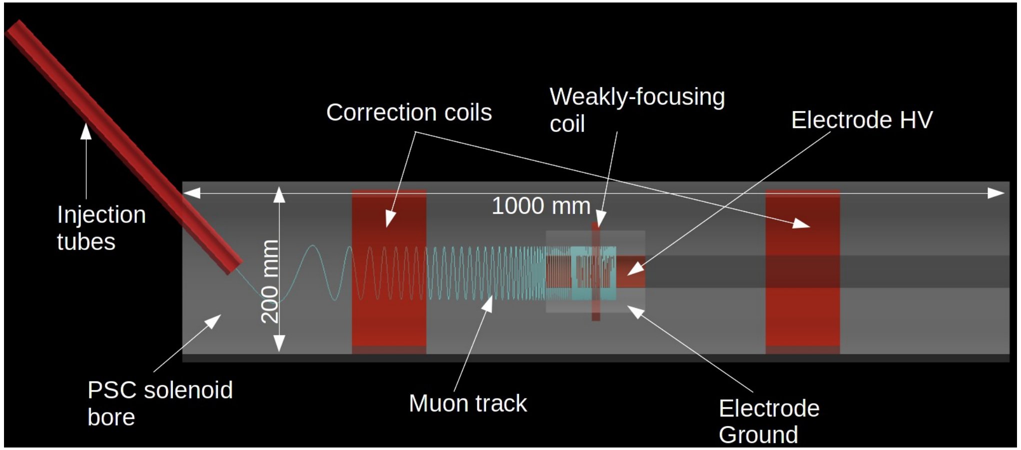
04.04.2024

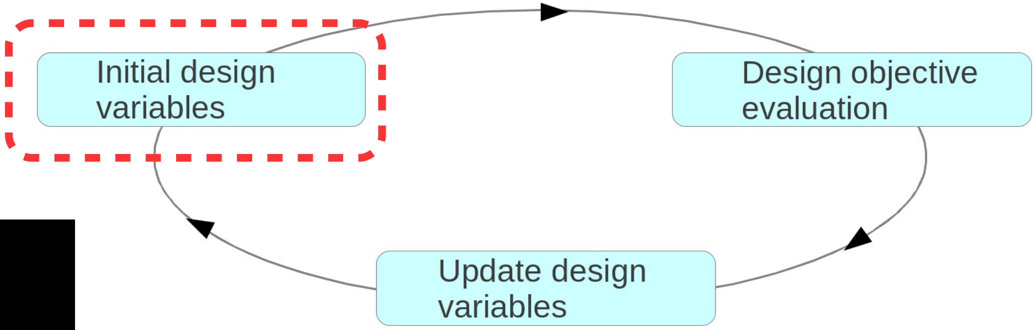
Optimization Layout



Optimization Layout

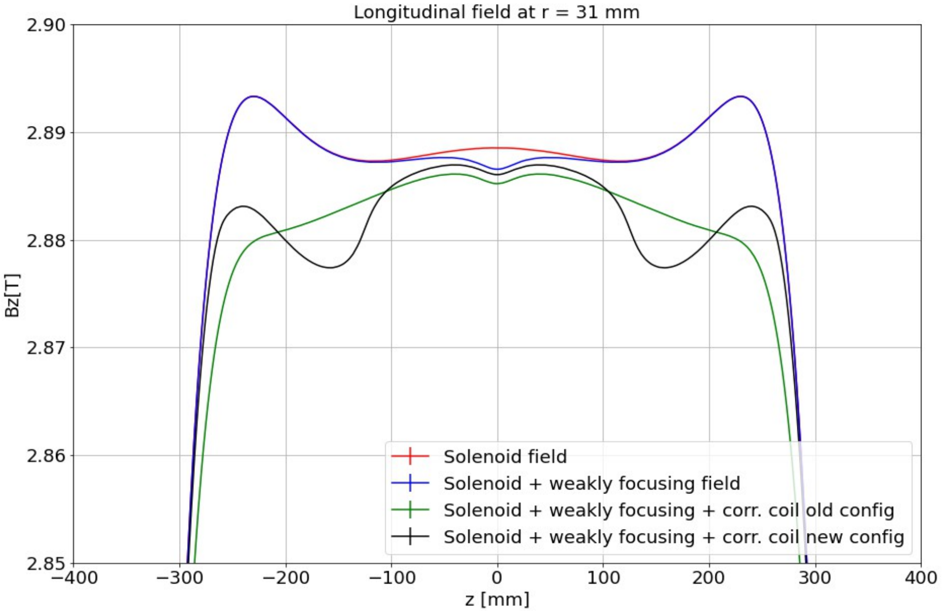
- Design simulation in g4bl





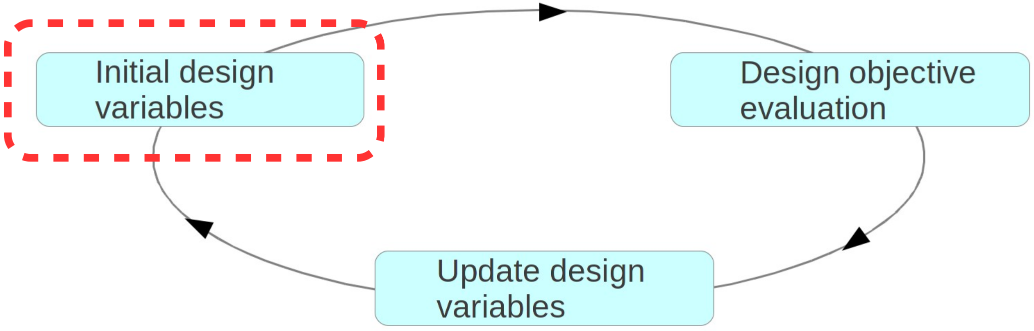
Design parameters:

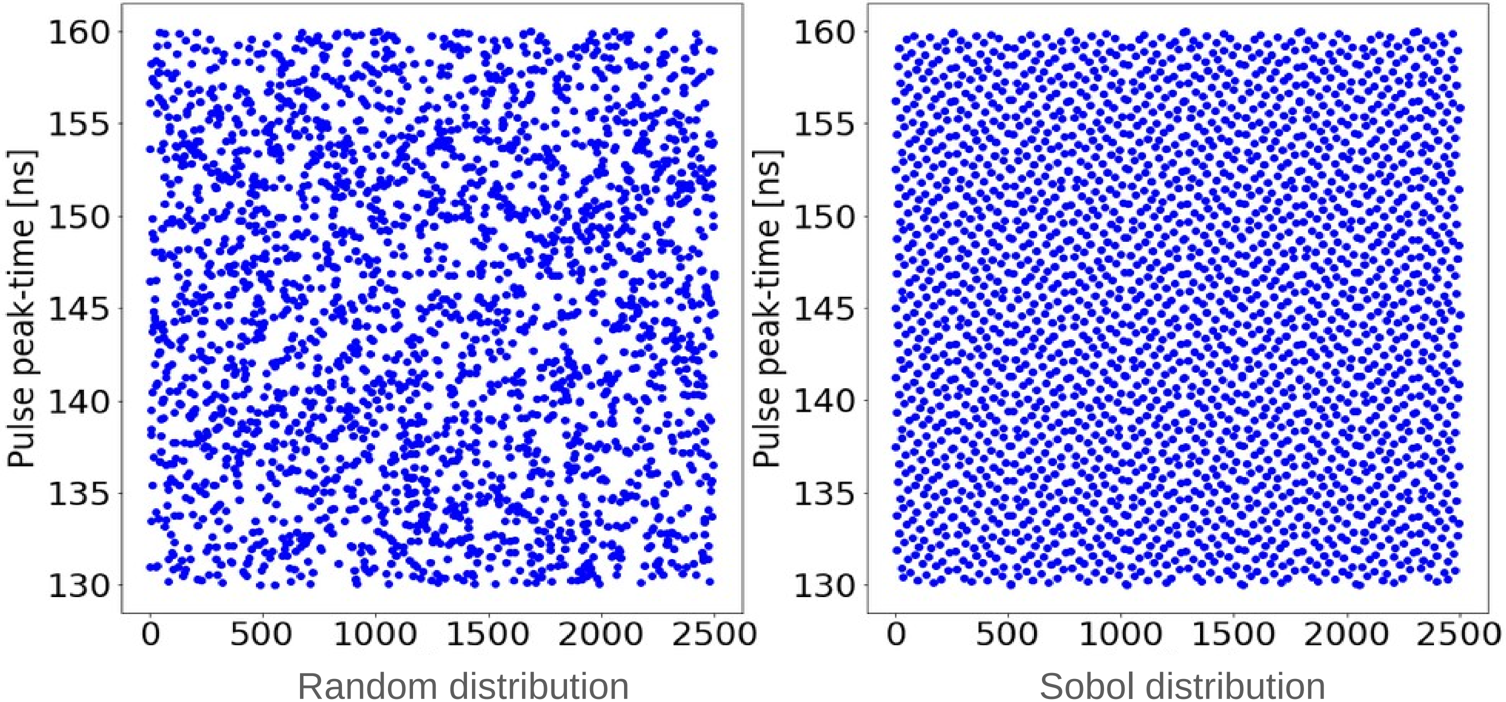
- Injection coordinates
- Magnetic field strength
- Correction coil features
- Weak-focusing coil features
- Kicker pulse features
- ……



Optimization Layout

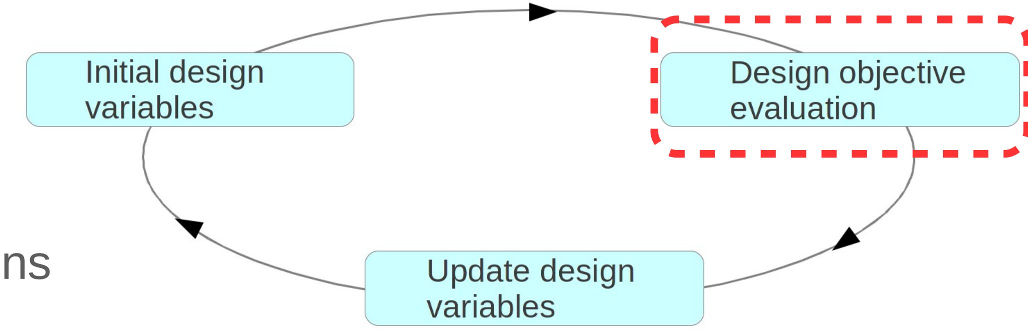
- Sampling input variables
- Sobol distribution
- Maximum uniform spread

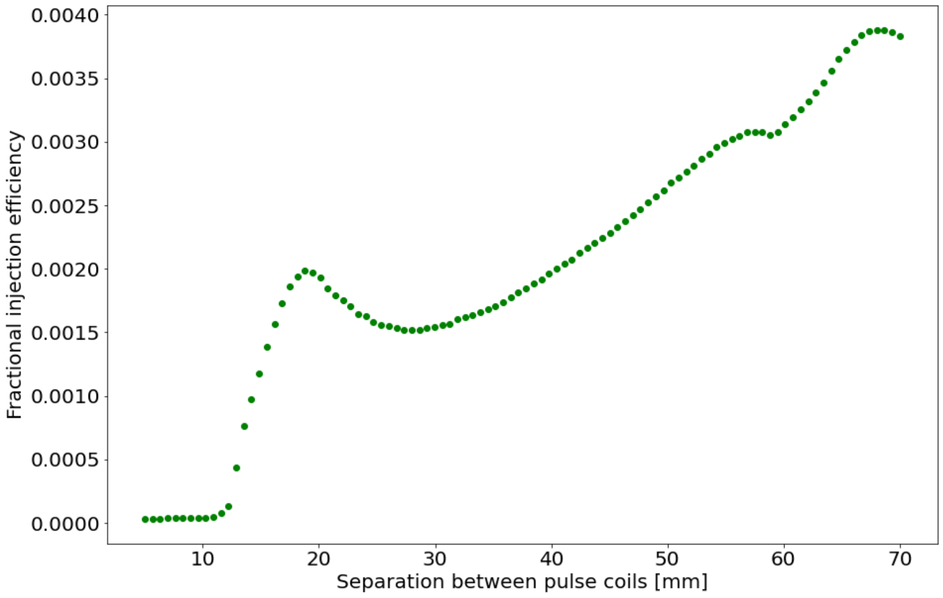


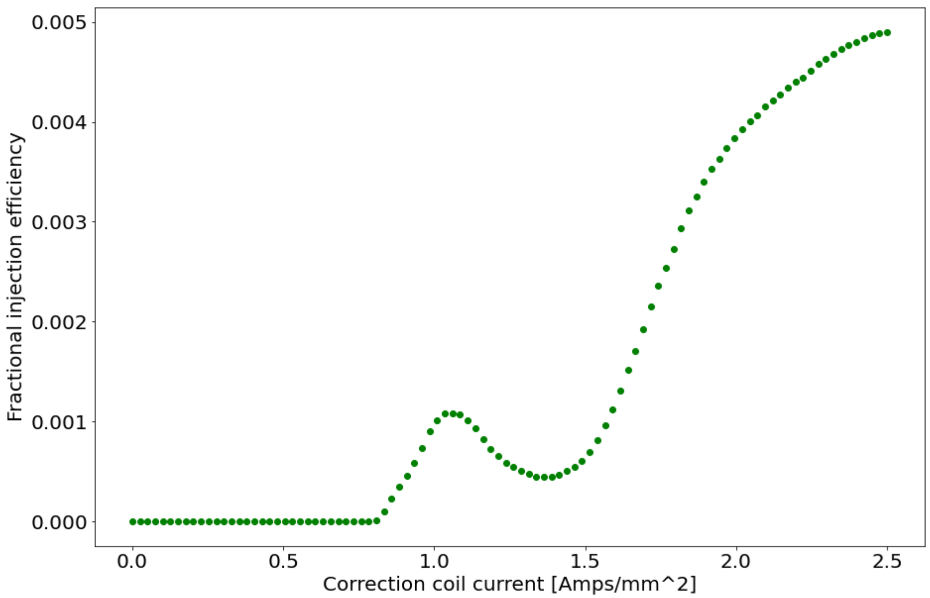


Optimization Layout

- Maximize injection efficiency
- Minimize power dissipation of setup
- Minimize polarization spread in stored muons
- ….







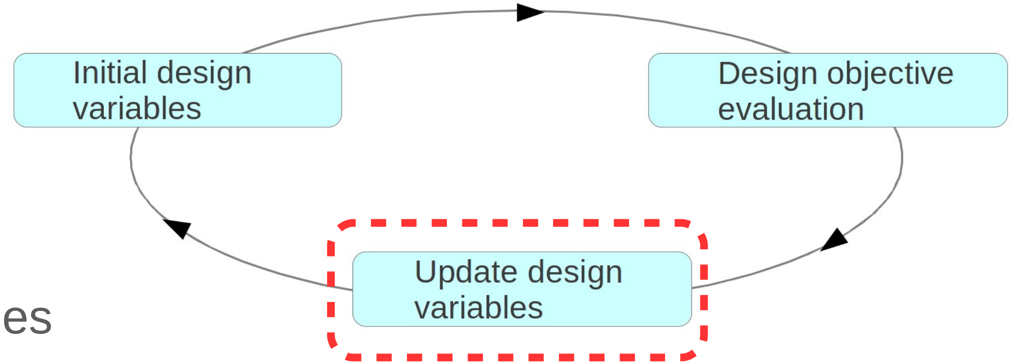
Optimization Layout

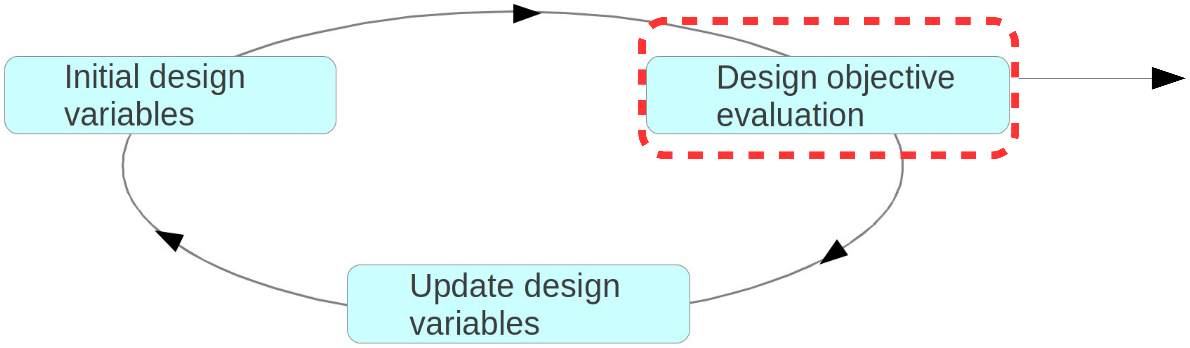
- Update design variables based on objective evaluation

- Repeat until optimal solution found

- Required to run simulation thousands of times
  → computationally expensive

- Replace physics simulation with approximation
  → surrogate model





Surrogate model for objective evaluation
→ Many ways
→ PCE and NN models explored

PCE Surrogate Model

- Polynomial Chaos Expansion (PCE) :

$$Y = \sum_{i=0}^{\infty} \alpha_i \Psi_i (\vec{x})$$

Y → Model response (injection efficiency), $\boldsymbol{\Psi_i}$ → Orthogonal polynomials

x → input variables, $\boldsymbol{\alpha_i}$ → expansion coefficients

- Polynomial basis based on input variable distribution

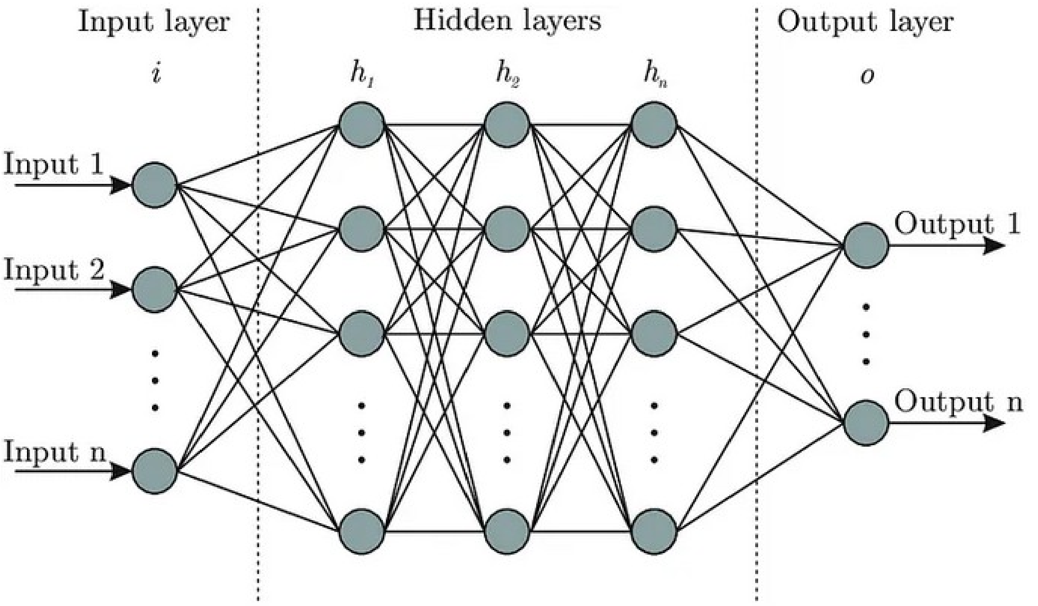
- Coefficients determined using regression based methods

$$\vec{\alpha} = \text{Argmin} \frac{1}{N} \sum_{j=1}^{N} \left\{ f(\vec{\xi}^j) - \sum_{i=0}^{P-1} \alpha_i \Psi_i (\vec{x}^j) \right\}^2$$

NN Surrogate Model

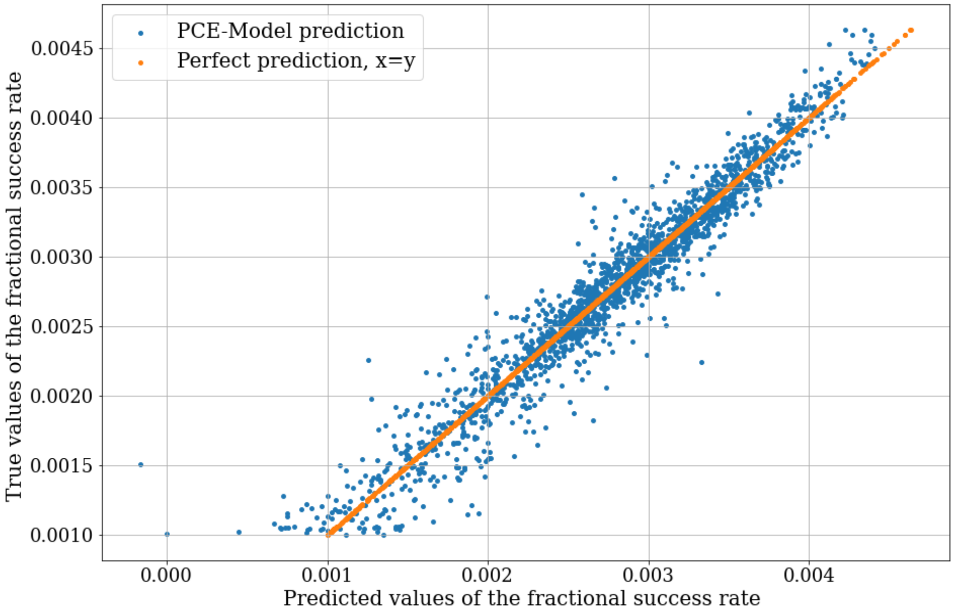
- Use the input (design) and output (objective) to train a neural network

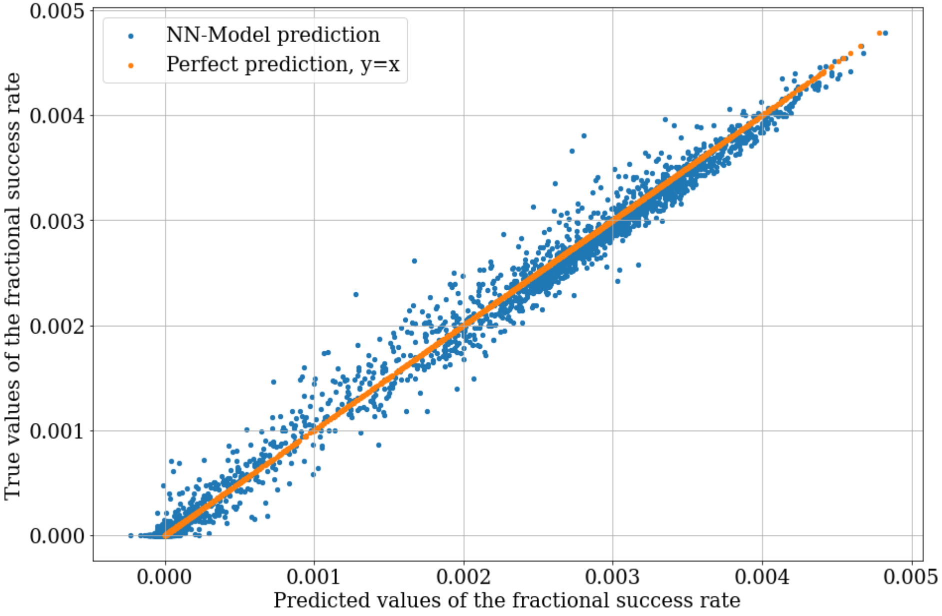
- Need to make choice for hyper parameters:
  → no. of hidden layers
  → no. of neurons
  → learning rate
  → optimizer
  → scheduler
  → activation functions
  → ….



Model Performance

Model performance for a 6 dimensional input space

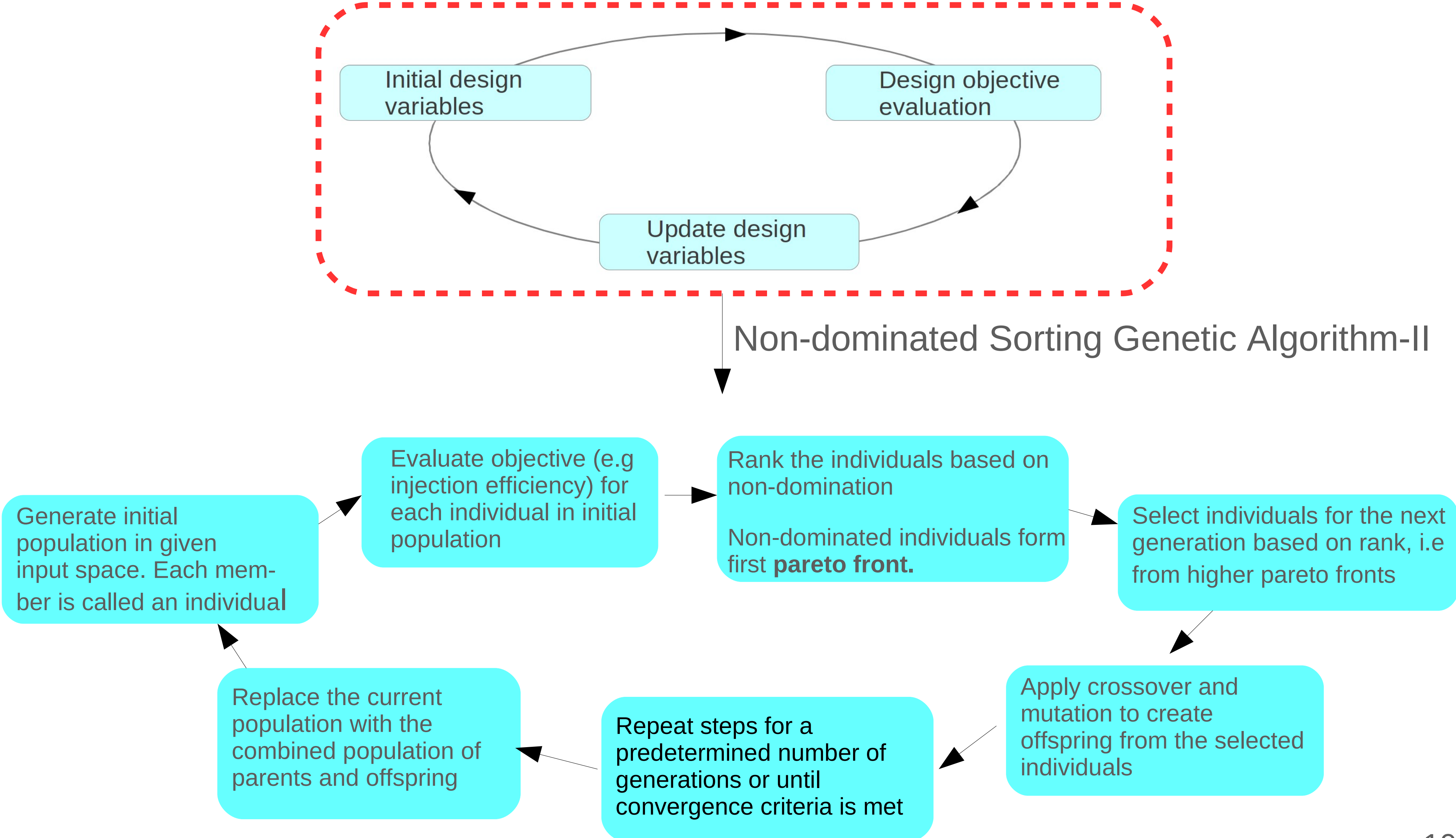




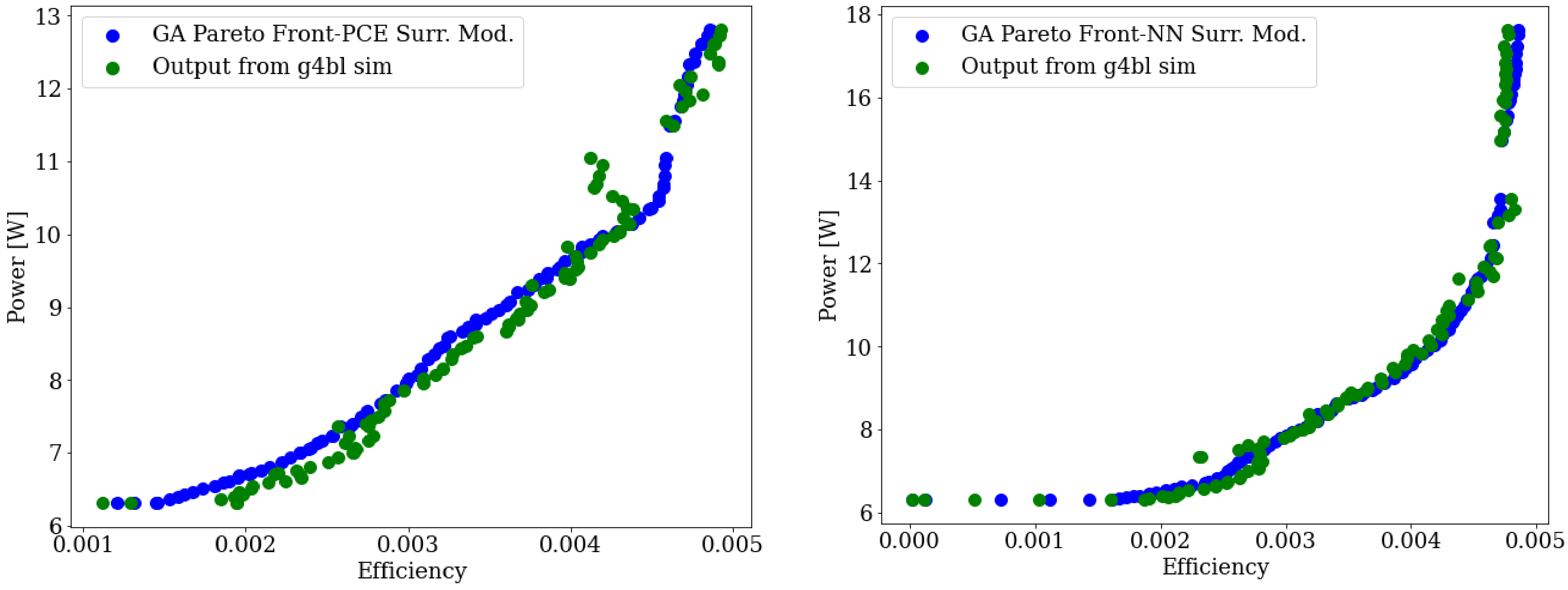
Mean Square Error: 3.47 e-08

Mean Square Error: 1.88 e-08

Genetic Algorithm: NSGA-II



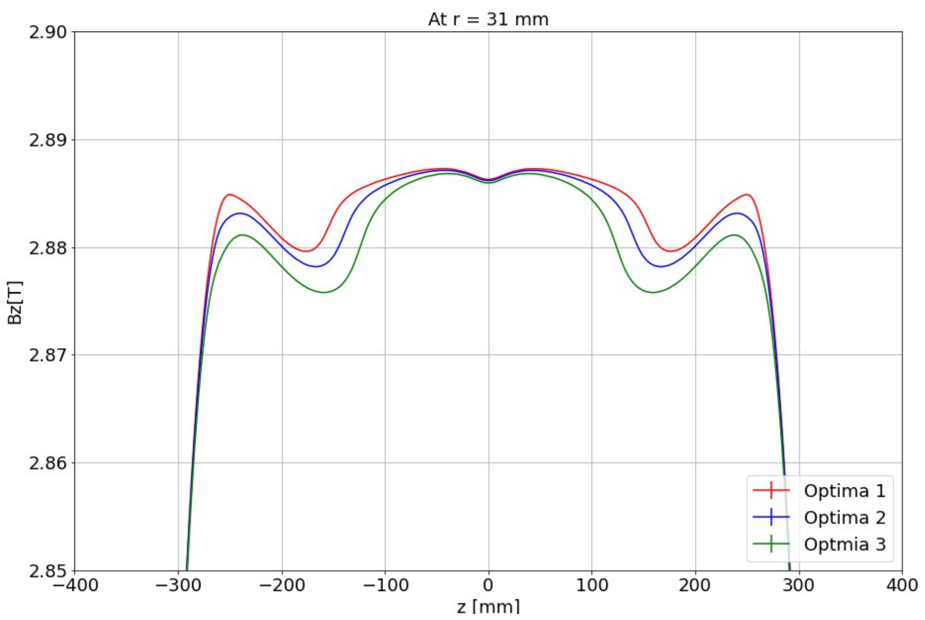
Surrogate model based NSGA-II performance

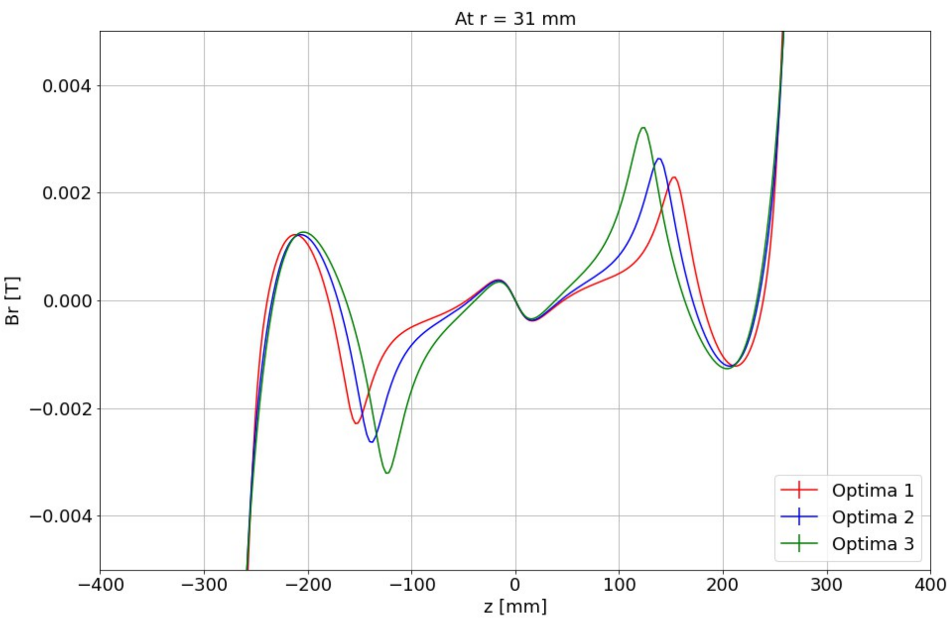


- 5% vs 2% agreement for PCE/NN based GA performance for average eff 0.35%

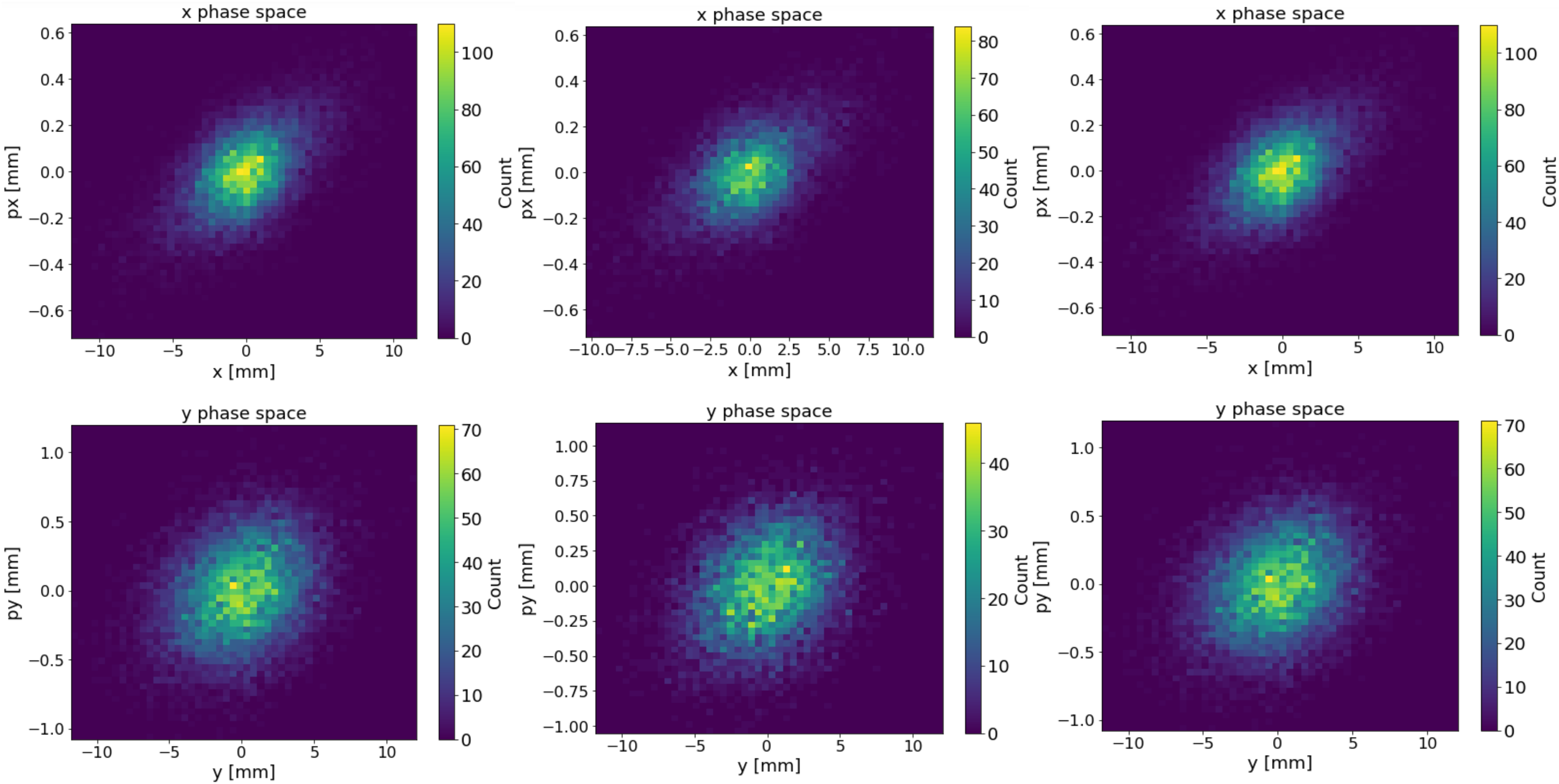
A look at Solutions: Magnetic Field Profile

	Efficiency	Kicker Timing [ns]	Kicker strength	Corr coil Length [mm]	Corr coil InnRadius [mm]	Corr coil OutRadius [mm]	Corr Coil Pos [mm]
Optima 1	0.0022	97.99	0.80	95.93	40.01	7.00	202.40
Optima 2	0.0032	91.80	0.72	118.31	40.01	7.10	198.90
Optima 3	0.0048	81.27	0.69	140.96	40.00	8.00	194.57





A look at Solutions: Storage phase space at Injection

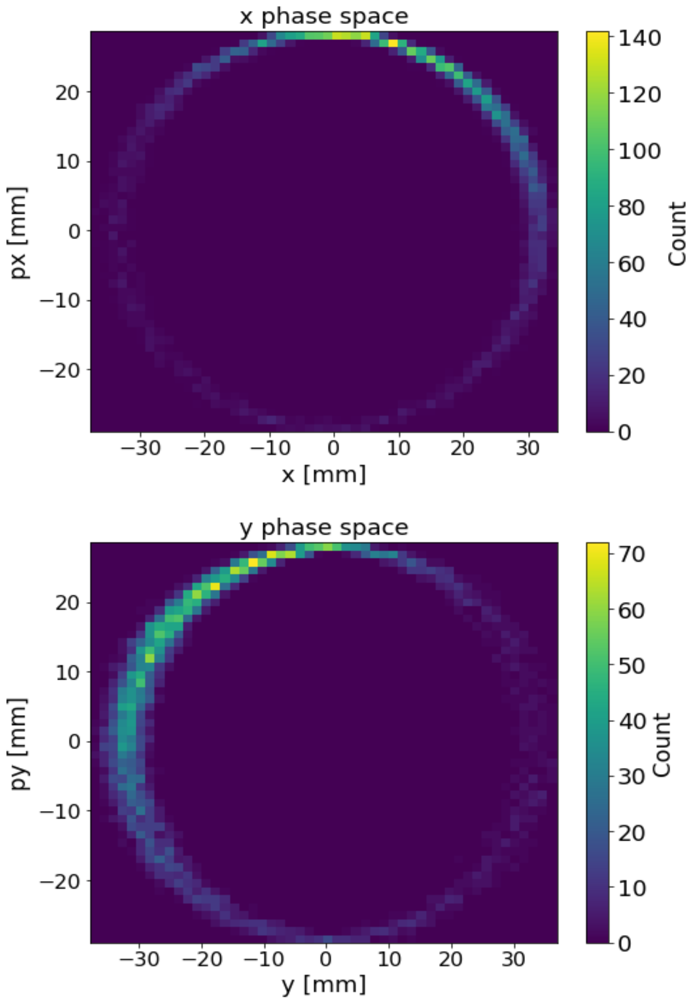


Optima 1

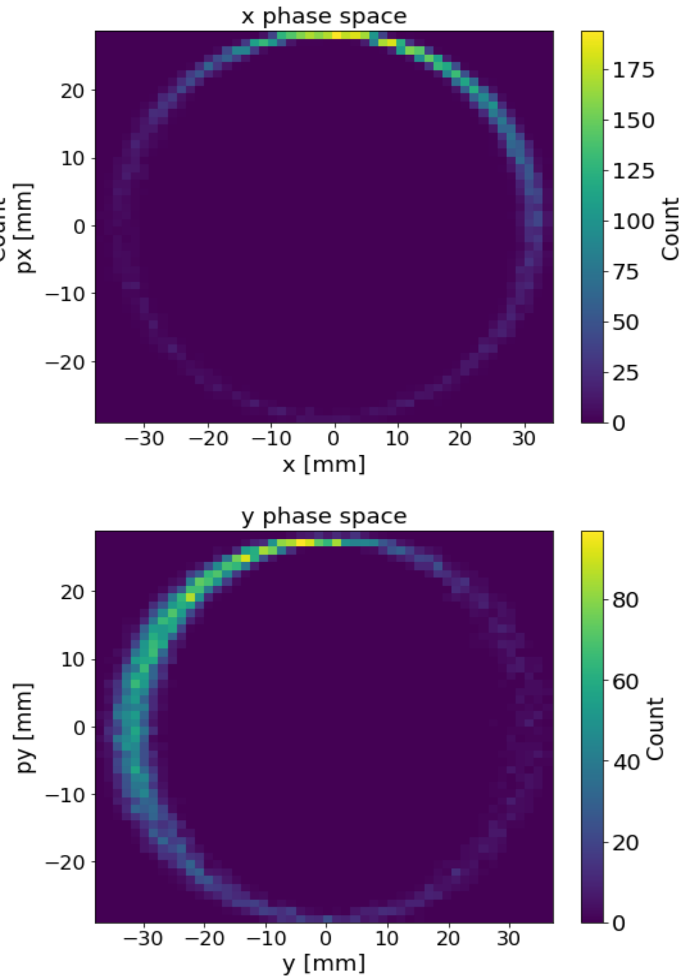
Optima 2

Optima 3

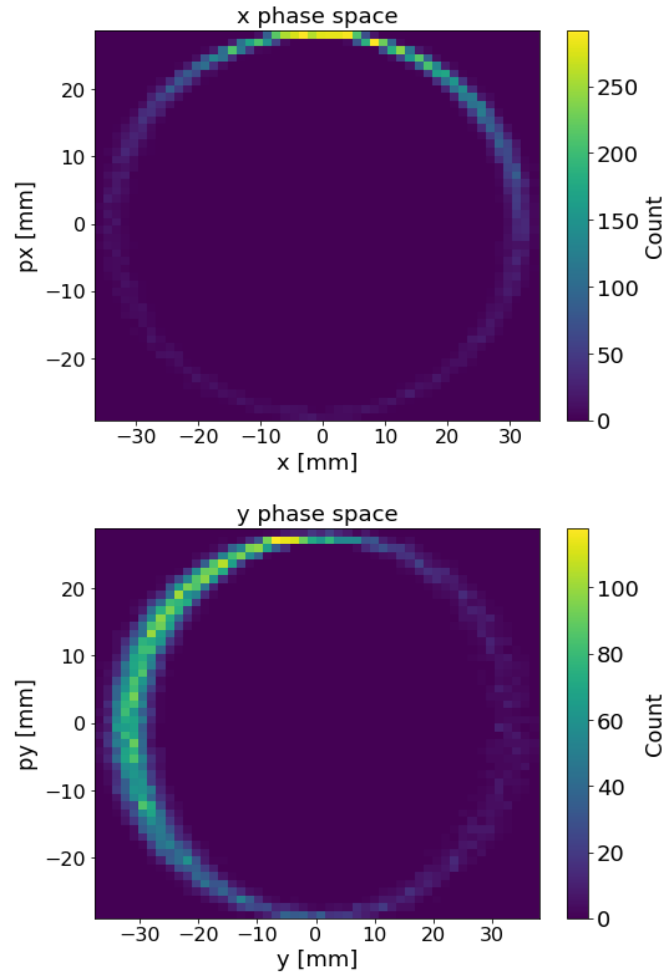
A look at Solutions: Transverse Storage Phase Space



Optima 1



Optima 2



Optima 3

Summary

- Running simulations iteratively is bottleneck in optimization process
- Orders of magnitude speed up can be achieved by replacing physics simulation by surrogate model
- Genetic algorithm NSGA-II used to run multiobjective optimization
- PCE and NN surrogate models based GA investigated; $\sim 10^3$ speed up for PCE, $\sim 10^4$ for NN
- No significant difference in storage phase space at injection or storage for 3 kicker timings
- Possibility to improve efficiency with bigger pulse coil separation
- Plan to expand into optmization where higher dimensional input space can be implemented with straightforward uncertainty quantification techniques

Acknowledgments

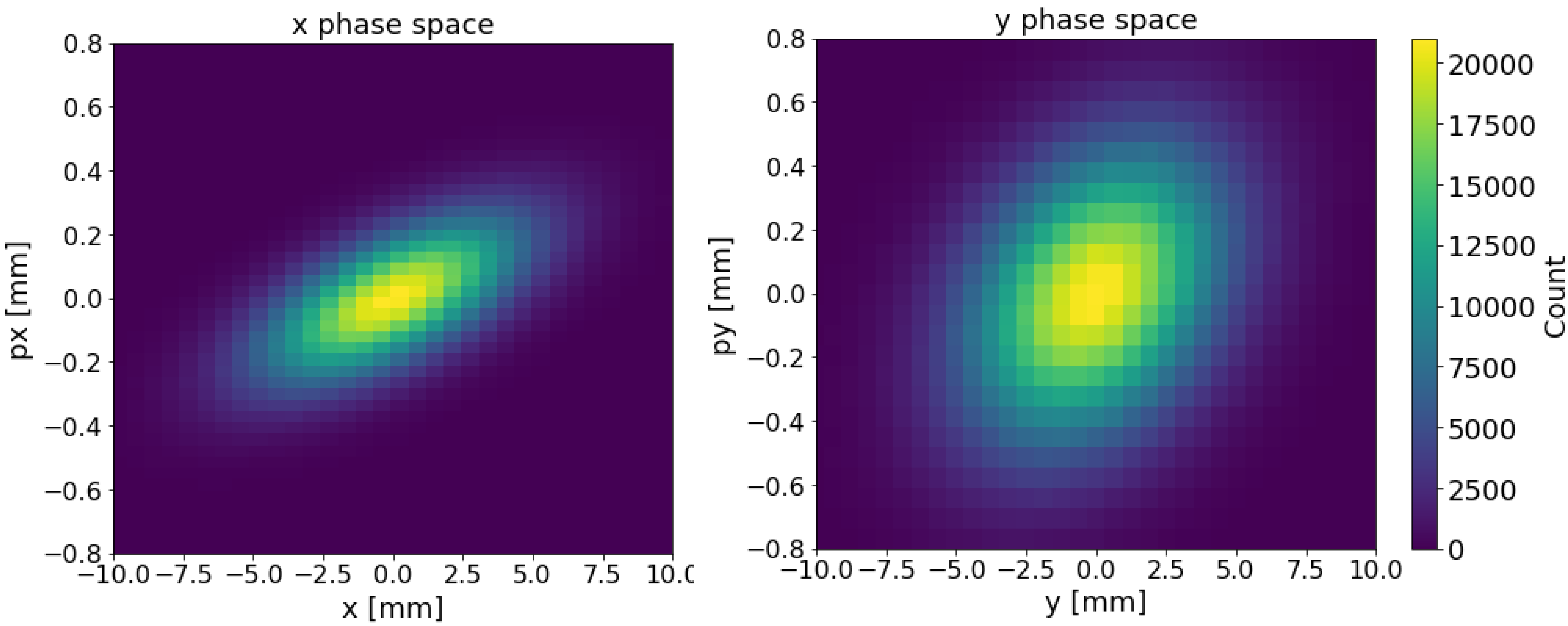
- Computational resources: PSI Local High Performance Computing cluster, Merlin6, Siyuan-1 cluster supported by the Center for High Performance Computing at Shanghai Jiao Tong University and the Euler cluster operated by the High Performance Computing group at ETH Zürich.

- Accelerator Modeling and Advanced Simulations (AMAS) group at PSI: A. Adelmann, S. Heinekamp and P. Juknevicius

- NN surrogate starting point: A. Holmberg Bachelor’s Thesis ETH Zurich 2021

Extra

Total phase space after collimation

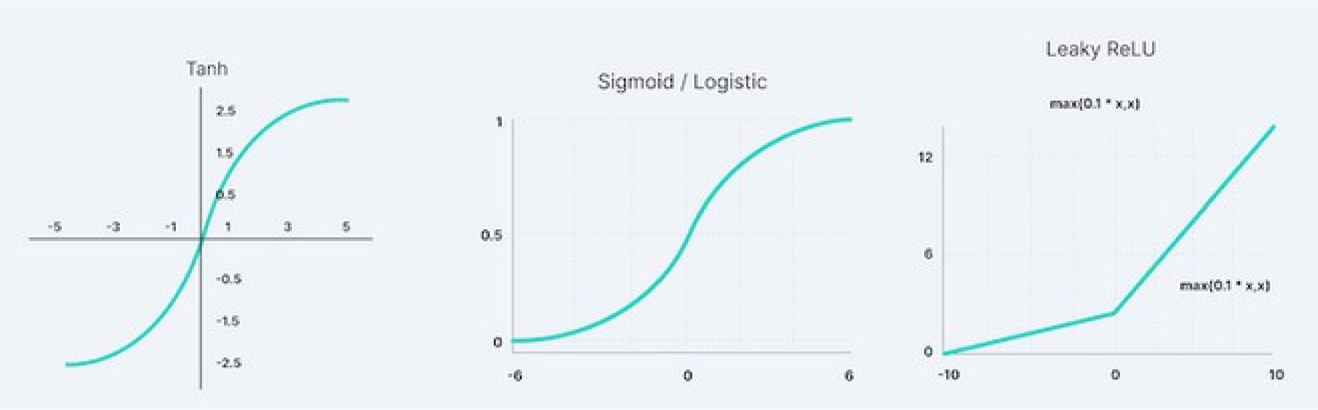


Neural Net hyperparameters

```python
def __init__(self, input_dimension, output_dimension, n_hidden_layers,
             neurons, regularization_param, regularization_exp):
    super(net, self).__init__()
    # Number of input dimensions n
    self.input_dimension = input_dimension
    # Number of output dimensions m
    self.output_dimension = output_dimension
    # Number of neurons per layer
    self.neurons = neurons
    # Number of hidden layers
    self.n_hidden_layers = n_hidden_layers
    # Activation function
    self.activation = nn.LeakyReLU()
    #
    self.regularization_param = regularization_param
    #
    self.regularization_exp = regularization_exp

    self.input_layer = nn.Linear(self.input_dimension, self.neurons)
    self.hidden_layers = nn.ModuleList([nn.Linear(self.neurons, self.neurons) for _ in range(n_hidden_layers)])
    self.output_layer = nn.Linear(self.neurons, self.output_dimension)

    self.dropout = nn.Dropout(0.1)
```

```python
# Model definition
my_network = net(input_dimension=x_train_norm.shape[1], output_dimension=y_train_norm.shape[1],
                 n_hidden_layers=8, neurons=512, regularization_param=0,
                 regularization_exp=0) #2  1e-5


# Random Seed for weight initialization
retrain = 134
# Xavier weight initialization
init_xavier(my_network, retrain)


optimizer_ = optim.Adam(my_network.parameters(), lr=1e-3)#, weight_decay=1e-5)
#optimizer_ = optim.LBFGS(my_network.parameters(), lr=0.1, max_iter=1,
#                         max_eval=50000, tolerance_change=1.0 * np.finfo(float).eps)

scheduler = optim.lr_scheduler.ReduceLROnPlateau(optimizer_, mode='min', factor=0.5, patience=500000)
#scheduler = optim.lr_scheduler.StepLR(optimizer=optimizer_, step_size=50, gamma=0.5)

n_epochs = 200
```

Neural Net activation function



6-d optimization parameter bounds

```
bounds = {"T_Offset": [80, 98],
          "BPI": [0.35,0.80],
          "CC_Len": [88, 150],
          "CC_Ir": [40, 84],
          "CC_Thick":[7,15],
          "CC_Pos":[166,241]}
```