

# CERN ACCELERATORS BEAM OPTIMIZATION ALGORITHM

E. Piselli<sup>1</sup>, A. Akroh<sup>1</sup>, K. Blaum<sup>2</sup>, M. Door<sup>2</sup>, D. Leimbach<sup>1,3</sup>, S. Rothe<sup>1</sup>

<sup>1</sup>CERN, Geneva, Switzerland

<sup>2</sup>Max Planck Institute for Nuclear Physics, Heidelberg, Germany

<sup>3</sup>Johannes Gutenberg University Mainz (IKP) Institute for Nuclear Physics, Mainz, Germany

## Abstract

In experimental physics, computer algorithms are used to make decisions to perform measurements and different types of operations. To create a useful algorithm, the optimization parameters should be based on real time data. However, parameter optimization is a time consuming task, due to the large search space. In order to cut down the runtime of optimization we propose an algorithm inspired by the numerical method Nelder-Mead. This paper presents details of our method and selected experimental results from high-energy (CERN accelerators) to low-energy (Penning-trap systems) experiments as to demonstrate its efficiency. We also show simulations performed on standard test functions for optimization.

## INTRODUCTION

Particle accelerators are, together with detectors, essential components for any experiment in nuclear and sub nuclear physics. Big accelerators are complex devices or machines that may serve many experiments at the same time.

Nowadays accelerator physics is a complex research field with applications far beyond subatomic physics. In order to deliver the best particle beams many adjustments are necessary. Controls plays a key role in all the technologies needed to run particle accelerators properly. All accelerator parameters should be controlled and adapted to the requested experiments. In this jungle of different setting configurations, algorithms for automated optimization can make a big difference on the time needed for setting up the machine. This saving of time is automatically translated into gain of operation time for the user.

Optimization is the discipline that deals with formulating useful models in applications, using efficient methods to identify the best possible solution. In mathematics, optimizing means finding the values which maximize or minimize a function.

Different approaches of finding optimal designs for a system are summarised in Table 1.

It is clear that the main advantages using optimization algorithms is due to the fast modelling and design process.

All processes in the accelerators are modelled and adjustments are made according to or with the help of these models. However, these models do not 100% represent the reality and therefore “intelligent” optimisation algorithms are vital to replace the often time consuming and handmade scans. The direct results are of course a gain in time and the enhanced performance of the machine, but we should

not forget that the outcome of these scan can help us to refine the models.

Researchers have developed many different algorithms using the most disparate methods like linear and gradient search methods, machine learning tools, genetic algorithm and many others. The algorithm we have developed and successfully tested is derived from the numerical method Nelder-Mead.

Table 1: Optimization Approaches Design for a System

Experimental optimization	Simulated models optimization	Optimization algorithm
<ul style="list-style-type: none"> <li>- Expensive</li> <li>- Tedious</li> <li>- Time consuming</li> <li>- Human involvement</li> <li>- Not always accurate</li> </ul>	<ul style="list-style-type: none"> <li>- Cheap</li> <li>- Slow design process</li> <li>- Medium human involvement</li> <li>- Error prone</li> </ul>	<ul style="list-style-type: none"> <li>- Fast modeling</li> <li>- Fast design process</li> <li>- Automated (minimum human involvement)</li> <li>- Low error</li> <li>- Complex optimization algorithm</li> <li>- Difficult of solving real world problems</li> </ul>

## BEAM OPTIMIZATION ALGORITHM

Direct-search methods do not use any information about derivatives and are therefore very robust with respect to small perturbations in the function’s values. Nelder-Mead [1] is the most known method within this group and the popularity of its practical applications is based on its simplicity.

The Nelder-Mead technique was invented by J. Nelder and R. Mead in 1965 as an evolution of the method of Spendley et al. [2]. From an initial suitable solution, the algorithm tries, at each iteration, to build an improved solution until the optimum is reached.

The idea is to define a simplex (polytope in  $n$ -dimensional space with  $n + 1$  vertices), each of which are connected to all other vertices (e.g. a triangle in  $\mathbb{R}^2$ , a tetrahedron in  $\mathbb{R}^3$ , etc.).

The initial simplex  $S$  is formed by  $n + 1$  vertices  $\vec{x}_0, \vec{x}_1, \dots, \vec{x}_n$  around an initial point  $\vec{x}_0 \in \mathbb{R}^n$ . The other vertices are created in order to get a full starting configuration:

Content from this work may be used under the terms of the CC BY 3.0 licence (© 2019). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI.

$$\vec{x}_k = \vec{x}_0 + h_k \cdot \vec{e}_i \quad \forall k \in [1, n]$$

where  $h_k$  is the step dimension in the direction of the unit vector  $\vec{e}_k$ .  $\mathbf{S}$  is a regular simplex where the length of all the sides of the geometric figure are equal.

For maximization, once the initial settings are defined, the Nelder-Mead loop follows these simple steps:

1. Ordering:
  - Find indexes of the best, the second best and the worst vertex in  $\mathbf{S}$ , respectively  $h, s, l$ :
    - $f_h = \max_k(f_k)$
    - $f_s = \max_{k \neq h}(f_k)$
    - $f_l = \min_{k \neq h}(f_k)$        $\forall k \in [0, n]$
2. Centroid:
  - Centroid calculation opposite to  $\vec{x}_h$ :
 
$$\vec{x}_c = \frac{\sum_{k \neq h} \vec{x}_k}{n}$$
3. Transformation:
  - New simplex is calculated using one of the four different operations according to continuous estimation of the function to optimise:
    - Reflection:
      - Reflection point  $\vec{x}_r = 2 \cdot \vec{x}_c - \vec{x}_h$
    - Expansion:
      - Expansion point  $\vec{x}_e = 2 \cdot \vec{x}_r - \vec{x}_c$
    - Contraction
      - Contraction point  $\vec{x}_{cont} = \vec{x}_c \pm \frac{1}{2} \cdot (\vec{x}_c - \vec{x}_h)$
    - Shrinkage
      - New points set  $\vec{x}_k = \vec{x}_h + \frac{1}{2} \cdot (\vec{x}_k - \vec{x}_h)$
      - $\forall k \in [0, n]$  and  $k \neq h$

The flow diagram of the Nelder-Mead algorithm for maximization is shown in Fig. 1.

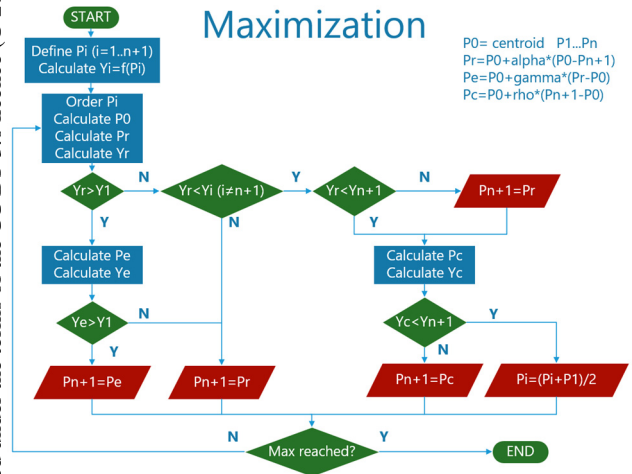


Figure 1: Nelder-Mead algorithm flow diagram for maximization.

The original algorithm [3], already tested at ISOLDE in 2012 [4], is used for unconstrained optimizations (an objective function that depends on real variables with no restrictions on their values). This was modified in order to fulfil our needs.

In the algorithm developed here, we have practically replaced the simplex points by the set of values of the  $n$

beam parameters to be optimized and the function by beam observables.

Beam parameters (variables) have to be defined in order to avoid scanning outside the region of interest. It is important to get proper limit settings for each parameter:

$$\vec{x}_{min} \leq \vec{x}_k \leq \vec{x}_{max} \quad \forall k \in [1, n].$$

Practically speaking, limits can be imposed through hardware if given by the device's working range or through software to limit the region of interest on a smaller range.

Once defined, the limit (variable constraints) the modifications carried out on our algorithm were the following:

- 1- Initial settings:
  - For each optimization scan, there is one starting simplex  $\vec{x}_0 \in \mathbb{R}^n$ . The others  $\vec{x}_1, \dots, \vec{x}_n$  vertices used for the first attempt are set to  $n - 1$  possible combinations of  $\vec{x}_{min}$  and  $\vec{x}_{max}$  limit vectors.
- 2- Constraints:
  - If during the optimization, the loop finds variable settings "outside" the box constraints, it retrieves a function estimation that is worse than the worst value found so far.
  - In this way, we try to "move" the simplex away from that parameters space region.
- 3- Convergence options:
  - In order to converge either to a target maximum or minimum  $f_m$  we need to define five parameters as inputs of the algorithm:
    - The minimum number of continuous iterations before convergence:  $it\_c$ .
    - The stability ratio:  $s\_r$  (in %).
    - The max convergence size:  $s\_c$ .
    - The automatic restart iteration number:  $it\_r$ .
    - The max number of iterations:  $it\_max$ .

Therefore, the convergence is obtained if the algorithm satisfies one of these criteria:

- Last  $it\_c$  evaluations are within a certain "envelope" with  $f_m$ :
 
$$|f_k - f_m| \leq s\_c \quad \forall k \in [n, n + it\_c]$$
- Last  $it\_c$  evaluations are stable within a defined stability ratio  $s\_r$ :
 
$$|f_k - f_{k-1}| \leq \frac{f_k \cdot s\_r}{100} \quad \forall k \in [n, n + it\_c]$$

It is clear that in the second criteria the algorithm may converge to a value, which is not always the value  $f_m$ .

- 4- Automatic restart:
  - After  $it\_r$  iteration if the optimization sequence does not converge to  $f_m$ , the algorithm restarts automatically using the last iteration value as starting point  $\vec{x}_0$ .
  - This automatic restart is performed not more than  $\frac{it\_max}{it\_r}$  times.

## SIMULATIONS

Different simulations have been performed in order to validate the robustness of the algorithm.

One of the analytic function used is the Rosenbrock function [5], which is defined as:

$$f(\vec{x}) = \sum_{k=1}^{n-1} 100 \cdot (x_{k+1} - x_k^2)^2 + (x_k - 1)^2$$

This function has a global minimum  $f(\vec{x}) = 0$  in a narrow parabolic valley  $\vec{x} = (1, 1, \dots, 1)$ .

We have tested the optimization algorithm with this function with a different number of variables and the same value range  $([-10, 10])$  for each parameter. We have applied random initial settings within the range.

Table 2 summarises some of the simulations performed with this function.

Table 2: Rosenbrock Simulation Results

Variables $n$	Evaluations for convergence
2	$\cong 80$
4	$\cong 400$
8	$\cong 1000$

Figure 2 shows Rosenbrock function in 2D and Fig. 3 shows the variables trend in this dimension.

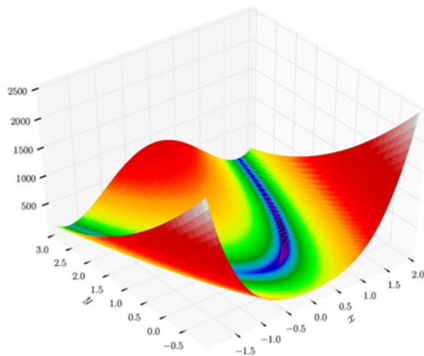


Figure 2: Rosenbrock 2D function [5].

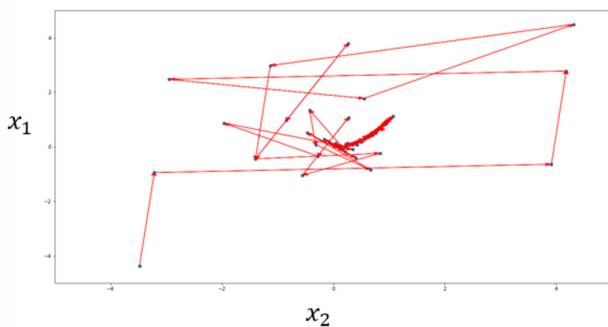


Figure 3: Rosenbrock 2D function variables optimization.

We have then tested the algorithm in the same condition as previously explained, but adding Gaussian noise with different  $\sigma$ . The results still showed good response, even though it did not converge to the minimum as shown in Table 3.

Table 3: Rosenbrock Simulation with Gaussian Noise Applied

Variables $n$	$\sigma$	Convergence values
4	0.001	$\cong [0, 0.1] \forall$ variable
4	0.01	$\cong [0, 0.4] \forall$ variable
8	0.001	$\cong [0, 0.2] \forall$ variable
8	0.01	$\cong [0, 0.5] \forall$ variable

## USE CASES

### ISOLDE

The ISOLDE Facility is dedicated to the production of a large variety of radioactive ion beams for different experiments in the fields of nuclear, atomic and solid-state physics, materials and life sciences. The facility belongs to CERN's accelerator complex.

Several automatic beam optimizations have been performed, especially in the low energy beam lines with particle energies up to 60 keV.

The aim of these optimizations was always to maximise the ion beam transmission through the machine, tuning electrostatic devices such as quadrupoles, steerers and deflectors.

Two examples are shown in Fig. 4 and Fig. 5, where we have simultaneously optimized 13 parameters and we measured a transmission gain of +225% and +16%, respectively.

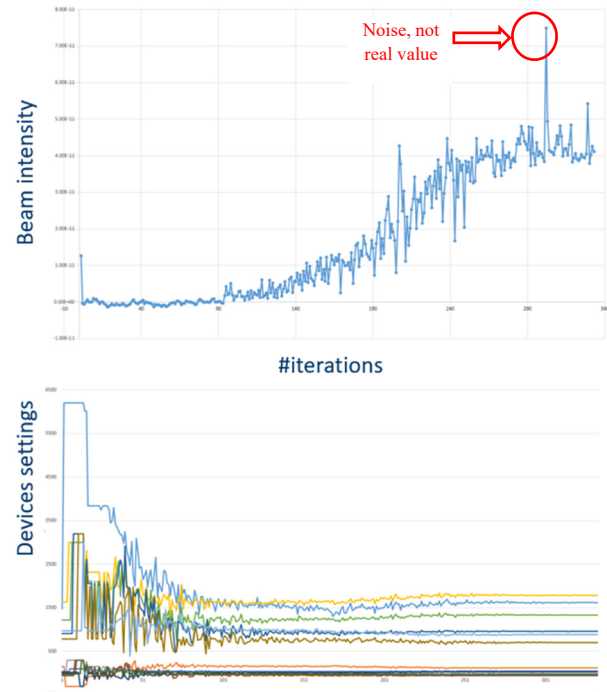


Figure 4: ISOLDE beam optimization with +225% gain (top: beam current intensity vs iterations bottom: devices values vs iterations).

Both optimizations required around 300 iterations (1.2 s interval, 6 min. total). Note that at each iteration all the values of the 13 devices have been changed at the same time.

Content from this work may be used under the terms of the CC BY 3.0 licence (© 2019). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI.

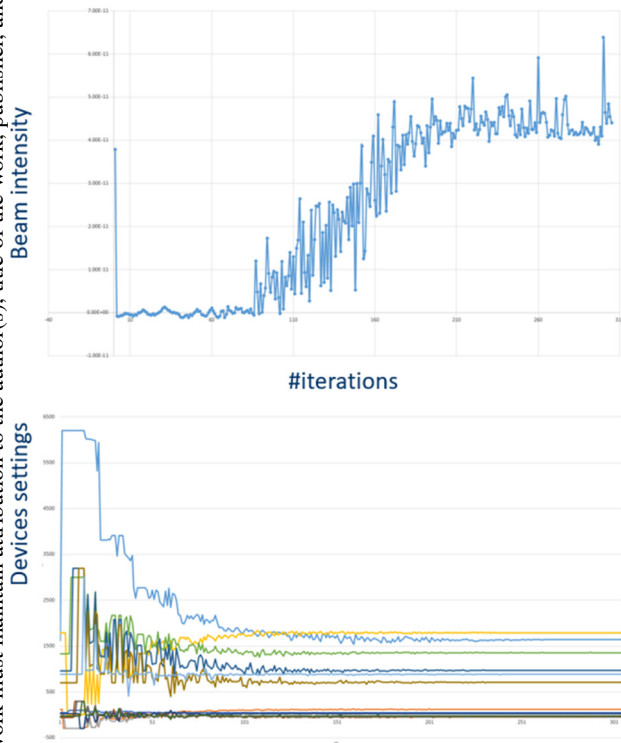


Figure 5: ISOLDE beam optimization with +16% gain (top: beam current intensity vs iterations bottom: devices values vs iterations).

Bottom pictures of Fig. 4 and 5 show the convergence of the devices values over time.

At ISOLDE, in 2018, we have run successfully about 300 automatic optimizations; the statistics are summarized in Fig. 6, 7 and 8.

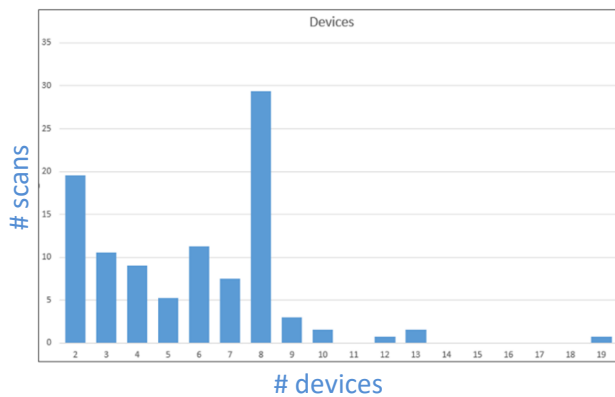


Figure 6: ISOLDE optimization statistics: devices optimised vs total optimizations.

It is interesting to notice that many successful optimizations have been performed with more than 6 devices and on average we needed 100-200 iterations to converge.

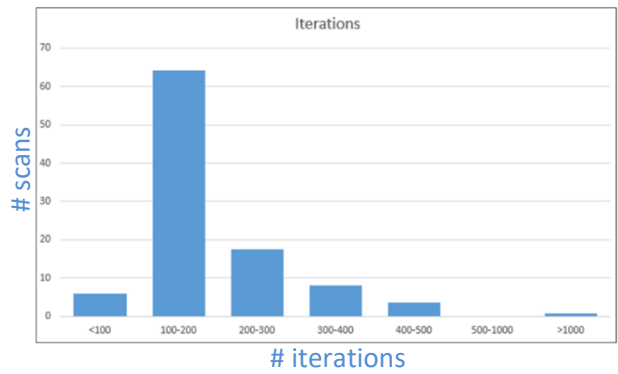


Figure 7: ISOLDE optimization statistics: total optimizations vs number of iterations.

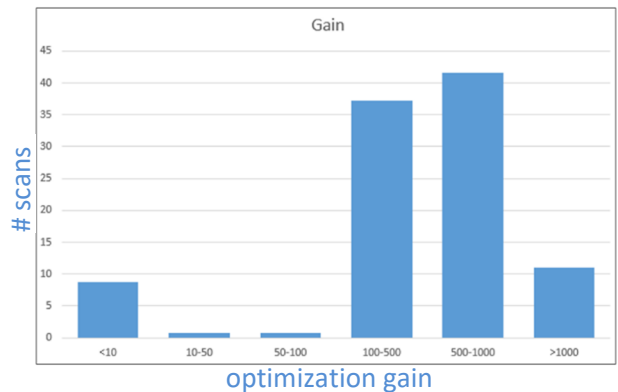


Figure 8: ISOLDE optimization statistics: total optimizations vs optimization gain [in %].

### GANDALPH

The GANDALPH experiment [6] (Fig. 9) aims at measuring the reaction between singly charged negative ions and a laser beam. The optimum tune for the experiment is achieved, when the ion beam is perfectly overlapped with the laser beam. The alignment is by two apertures of 6 mm diameter placed at a distance of 500mm along the center of the beamline.

In order to guide the ion beam through these apertures, a pair of horizontal kicker plates, and two x/y steerer boxes are used. These electrostatic elements are part of the experimental beam line and the applied voltages can be computer controlled through an independent suite of applications, implemented in LabVIEW. Previously, the Optimizer has been used to optimize the injection into the experimental beamline in an iterative manner; alternating between automatically tuning ISOLDE elements and manually tuning the experimental beamline. The signal on which one could optimize was previously limited to ion beam current via the ISOLDE Faraday cup acquisition.

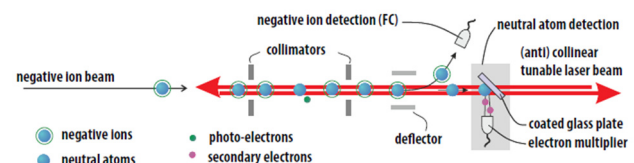


Figure 9: GANDALPH beamline layout (schematic).

We have now successfully demonstrated the simultaneous optimization of ISOLDE parameters and experiment parameters using a user-generated signal (event rate from an electron multiplier). For this, we have written an interface that enabled us to set parameters of the user's electrostatic elements via a LabVIEW Common Middle Ware (CMW) wrapper [7]. We have created a virtual device in the CERN Controls Database, that contained eight control parameters and one device that represented the signal.

This prove of concept was rather reliable and has been successfully used during the on-line campaigns of the GANDALPH detector. In Fig. 10 an example of the optimization is shown.

The flexibility of CMW and the available wrappers will allow us in future to open the Optimizer software as a part of the infrastructure to the ISOLDE user community through interfaces using JAVA, Python or LabVIEW.

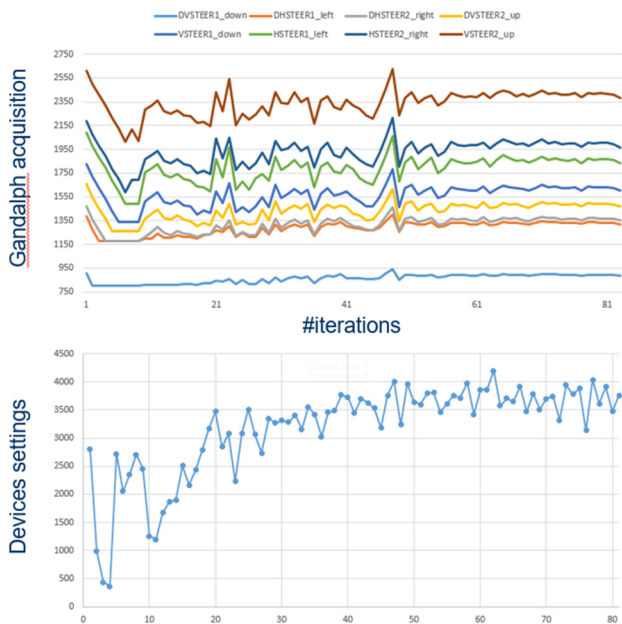


Figure 10: Gandaliph optimization, for details see text.

### Penning Trap

Penning-trap experiments dependent on external ion sources with typically low energy ion beams (about a few keV). Specifically the usage of pulsed drift tubes for deceleration and the very small acceptance of the storage magnet pose difficulties during beam optimization. Pulsed drift tubes have to be optimized in timing and need subsequent optimizations with focusing lenses due to their negative effect on the beam emittance. In combination with the small acceptance of the trapping system in the superconducting magnets, the need for an optimization system capable of working with a very low signal in most of the parameter space is high.

For on-line experiments with limited beam time a fast algorithm for beam optimization would directly contribute to better statistics for measurement data and also for offline measurements the Optimizer would spare time on

preparation and could increase the number of measurements per time.

At an example optimization at Pentatrap (Fig. 11) [8] with a 6.5 keV pulsed beam of highly charged ions, the potentials on the third electrostatic steerer/lens combination (einzel lens 3), electrostatic lenses 1-3 and the electrostatic quadrupole bender were optimized (in total eight device parameters) using the response signal of a Faraday cup at the end of the trapping system. The initial device parameters were chosen randomly inside a reasonable range. In order to get a representative signal for the given settings the trigger for the beamline was delayed by a few seconds after new parameters have been applied and the pulse intensity signal on the Faraday cup was averaged over multiple shots.

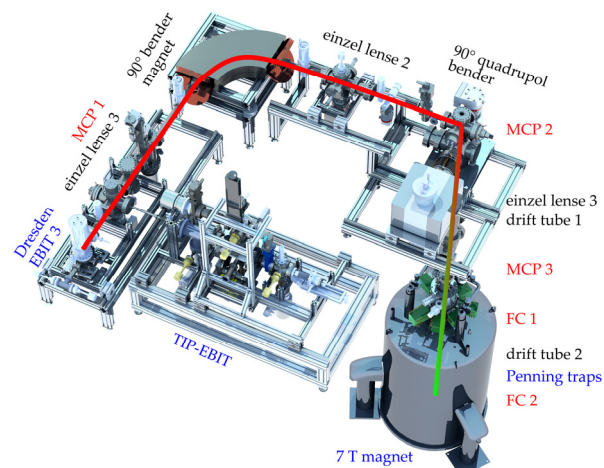


Figure 11: Pentatrap system [8].

Using the Optimizer tool, the beam performance could be increased from 7.5 a.u. to 9.8 a.u., corresponding to +30 % in only 96 iteration steps.

## CONCLUSIONS AND PERSPECTIVES

The algorithm has been fully tested and is now very robust. It showed important improvements in the operations of different experiments. The ongoing collaboration with Max Planck Institute for Nuclear Physics in Heidelberg and different CERN groups will significantly improve the functionality and usability of this tool from high-energy to low-energy physics experiments.

We foresee to add noise reduction filtering, based on real time averaging and to improve the automatic loop restarting to be able to explore different regions of interest.

## ACKNOWLEDGMENT

We would like to thank ISOLDE collaboration, CERN BE-OP management and MPI for Nuclear Physics Heidelberg for their full support.

## REFERENCES

- [1] Margaret H. Wright, "Nelder, Mead, and the Other Simplex Method", Documenta Mathematica Extra Volume ISMP (2012) 271–276.

Content from this work may be used under the terms of the CC BY 3.0 licence (© 2019). Any distribution of this work must maintain attribution to the author(s), title of the work, publisher, and DOI.

- [2] Spendley W., Hext G. R., "Sequential Application of Simplex Designs in Optimization and Evolutionary Operation.", *Technometrics* 4, 441-461, 1962.
- [3] E. Piselli, A. Akroh, "New CERN Proton Synchrotron Beam Optimization Tool", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 692-696. doi:10.18429/JACoW-ICALEPCS2017-TUPHA120
- [4] T. Giles, E. Piselli, "Automatic tuning of complex beam-lines", 16th International Conference on Electromagnetic Isotope Separators and Techniques Related to their Applications (EMIS2012), Matsue, Japan.
- [5] <https://www.sfu.ca/~ssurjano/rosen.html>
- [6] S. Rothe *et al.*, "Laser photodetachment of radioactive<sup>128</sup>I<sup>-</sup>", *J. Phys. G: Nucl. Part. Phys.* 44 (2017) 104003 (10pp).
- [7] O. O. Andreassen, D. Kudryavtsev, A. Raimondo, A. Rijllart, S. Shaipov, and R. Sorokoletov, "The LabVIEW RADE Framework Distributed Architecture", in *Proc. 13th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'11)*, Grenoble, France, Oct. 2011, paper WEMAU003, pp. 658-661.
- [8] J. Repp *et al.*, *App. Phys. B* 107, 983 (2012).