Linac 4 beyond classical control

Linac4 HW and BC WG Meeting

S. Hirlaender, V. Kain, B. Mikulec, A. Akroh, A. Lombardi, S. Tomin, I. Agapov…
Motivation

- Improve reliability, efficiency of Linac4 where classical controls are not available or working:
  - Model not available in the control room
    - LINACs do not fit (yet) into LSA style online models, plus there is space charge
  - Arduous translation from the simulation to the application.
  - But: no need for manual optimisation anymore ⇒ it can be done by the computer faster, better and reliably (no matter who launches the algorithm - same result).
• Try to minimise the number of interactions to reach the objective - the goal for all methods - sample efficiency.

• If there is the possibility to use a model, it must be used.

• In real applications: there are usually bounds and constraints.
Applications to LINAC4 chopping
Overview machine

- **H⁻ source**

- Repetition rate 0.833 Hz (one shot/BP)
Example 1: chopping efficiency optimisation

- Pulsed electric field to give Transverse RF kick to remove unwanted beam @ 3 MeV
- Loss-free injection into the PSB (4 rings) and into the PSB RF bucket (~ 630 ns)
- Remove beam during the rise time of PSB distributor
- Produce beam segments per ring at PSB revolution frequency

Chopping efficiency depends on optics between chopper and chopper dump
Goal

- Optimise the chopping pattern using distance to reference ($\chi^2$).

- No beam to DTL:
  - 100 % of beam to be chopped.
  - No safeguard from transmission $\Rightarrow$ no guarantee where beam will be lost

- Hole in bellows before chopper

→ Optimisation with watchdog and constraints
Automatic optimization of transmission and chopping efficiency MEBT

- Goal: optimise optics to fully chop beam (lose it on chopper dump: ensured by vacuum signals)
- Vacuum signals can be used like BLM signals
- Integrated in the optimization environment
- Reduced intensity during tests: ~ 15 mA
  - In view of optimising chopping pattern and transmission in the future: observable $\chi^2$ between reference pattern and measured pattern!

- Several new algorithms were successfully tested, including Bayesian optimization (model building)
- Time to find the optimum is reasonable (5 DOF - several iterations)
Constrains - avoiding damage
- or keeping performance

\[
\min_{x \in \mathbb{R}^n} f(x)
\]

\[
\text{s.t. } 0 \leq g_i(x), \ i \in \{1, \ldots, m\}
\]

- Here are two thresholds:
  - A soft threshold \( g_i \)
  - A hard threshold on the machine \( g_i \) a-priori unknown - measured!

- We take the feedback as a constrain. Either from a simulation or the machine (vacuum level).
Many algorithms in our toolkit...

- There are adequate optimization algorithms available to solve specific cases:
  - Fast and convex with constraints: COBYLA (not simplex)
  - Model building (reusable) - Bayesian optimization
  - Stabilising: Extremum seeking
Applications to LINAC4 steering
LINAC 4 layout and overview

- Repetition rate 0.833 Hz (one shot/BP)
Example 2: trajectory steering

Trajectory steering along LINAC after LEBT
- 17 BPMs H&V
- 16 correctors H&V

Reduced pulse length to ~ 50 us and current to ~ 5 mA
OCELOT@LINAC4

OCELOT

- Opensource framework for online beam dynamics simulation and optimization. Development started by S. Tomin (European XFEL) and I. Agapov (DESY)
  - Was introduced and recommended at both ICFA ML workshops

- Example: simplex optimization with 5 correctors

Not able to achieve satisfying results given this short amount of time
Let’s optimise ;)  
COBYLA - Just do it!

Horizontal plane 16 DOF ~ 70 interactions

![Graph showing Actors and Objective](image-url)
Measured response matrix →
but validation failed…

Was not reproducible!
A self-learning controller?

Reinforcement learning

- A controller is reading a state $s$ and proposes an action $a$:
  - $\pi(s) \mapsto a$
- $\pi(s)$ can be arbitrarily complicated - non linear.
- The controller learns this mapping from interaction with the environment autonomously and optimises the objective.
- After the training the controller can solve the problem in only one - a few steps - (like YASP, but with arbitrarily non-linear model).
Training the self-learning controller

- An episodic training.
- Actions $a$: 15 DOF (horizontal magnets).
- States $s$: Horizontal BPMs.
- Goal $r$: Minimise the RMS.
- If a threshold is hit, the training restarts with random initial conditions.
- The controller learns and gets faster to reach the threshold.
- The controller learns directly from interactions.
- **We use a highly sample efficient self-implemented algorithm which can be obtained via pip install pernaf - an improved version of the normalised advantage function algorithm.**

Impressive result to be published!
The training metrics

• The controller converges after ~250 steps.

• Green curve shows the expected minimal RMS by the algorithm - indeed the best reachable value.

• 1. Successful test of deep-reinforcement-learning @ Linac4
Future
Model based approach
From the model to the real world - AWAKE

- The AWAKE transfer line is very similar to the Linac4 steering test case - **difference we have a model!**
- 10 DOF - magnets to steer electron beam, 10 BPM to measure the trajectory, RMS to be minimised.

Training of the controller on the simulation

Applied on the machine. Residual physics could be trained directly with a **small number of interactions**
No model?

Learn it!

- Develop uncertainty aware data driven model ensemble of Bayesian neural nets.

- Train the controller on this model.

- Model updated automatically if controller not good enough.

Offline available?

- Collect data
  - Train the dynamics model
  - Train any controller on the model - ensemble
  - Test on real environment
  - Collect more data to improve model following a π(s)

Happy?

Done!
**Model based - model free**

Highly sample efficient algorithm learning the controller directly PER-NAF

Assumes specific problem structure

Highly sample efficient algorithm learning the controller indirectly on a model ensemble

Ability to overcome limitations while being sample efficient

---

Model free-
Here on the machine

Model based -
Here on simulation
Final remarks

• Ideal approach: develop online model for LINAC4 → needs new ideas. So far only synchrotrons.

• Even if there is online model, not necessarily correction algorithms available. E.g. optics matching

• Gained confidence and experience in the last two years with automatic tuning and optimisation algorithms and machine learning for model preparation.
  
  • LEIR, SPS, PS, PSB, AWAKE, LINAC4,…
  
  • No justification for lengthy manual tuning → algorithms are better than us.

• Growing community at CERN with expertise in advanced optimisation and control algorithms → ML coffee

Lot’s of potential at LINAC4
Resources - transparency

- PER-NAF or `pip install pernaf`
- Linac4 tests
- NAF AWAKE tests
- Model ensemble tests
- Questions: simon.hirlaender@cern.ch, verena.kain@cern.ch
Thanks for your time
NAF

- NAF very sample efficient Q learning algorithm

$$Q_\phi(s, a) = -\frac{1}{2}(a - \mu_\phi(s))^T P_\phi(s)(a - \mu_\phi(s)) + V_\phi(s)$$

![NAF Architecture](image)

$$\arg\max_a Q_\phi(s, a) = \mu_\phi(s) \quad \max_a Q_\phi(s, a) = V_\phi(s)$$