14th International Particle Accelerator Conference,Venice, Italy JACoW Publishing
ISBN: 978-3-95450-231-8 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2023-MOPL112

MAD-NG FOR FINAL FOCUS DESIGN

E. Manosperti*, L. Deniau, J. M. Gray, R. Tomás, A. Pastushenko, CERN, Geneva, Switzerland

Abstract

The CLIC Beam Delivery System (BDS) transports the lepton beams from the exit of the Main Linac to the Interaction Point (IP). The Final Focus System (FFS) is the last part of the BDS and its role is to focus the beam to the required size at the IP and to cancel the chromaticity of the Final Doublet (FD). MAD-X and MAD-NG are simulation codes for beam dynamics and optics that are used for particle accelerator design and optimization. This paper presents a comparison between the two codes to achieve the best performance of the design of the FFS, including the optimization methods, the speed performance, and the physics accuracy.

INTRODUCTION

MAD-X [1–3] and MAD-NG [4, 5] are simulation codes for beam dynamics and optics used for particle accelerator design and optimization, both open-source. MAD-X was released in 2002 and it is based on the programming languages C, C++, Fortan77, and Fortran90. MAD-NG has been developed since 2016 in parallel to MAD-X, and is based on the Lua programming language [6] with few extensions, and LuaJIT [7], the tracing just-in-time compiler for Lua. Its physics is based on the symplectic integration of differential maps made out of Generalized Truncated Power Series Algebra (GTPSA) [8]. This paper reports a comparison between the latest release of the two codes, MAD-X 5.8.1 and MAD-NG 0.9.6, to achieve the best performance for the CLIC FFS design [9] for an energy of 7 TeV [10] in the center-of-mass.

TWISS

The *Twiss* [1,4] command provides a single interface to calculate the linear lattice functions.

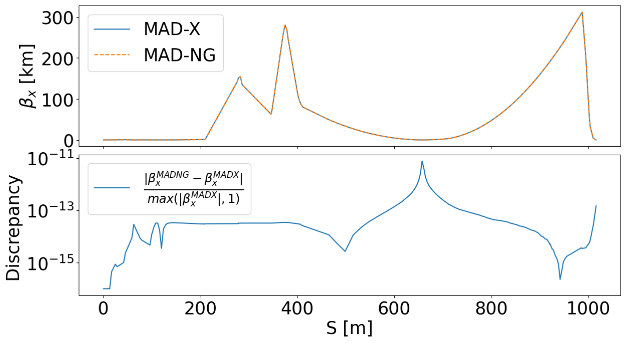


Figure 1: Horizontal β function along the FFS computed with MAD-X (blue) and MAD-NG (orange) (top). Discrepancy between the MAD-X and MAD-NG outputs (bottom).

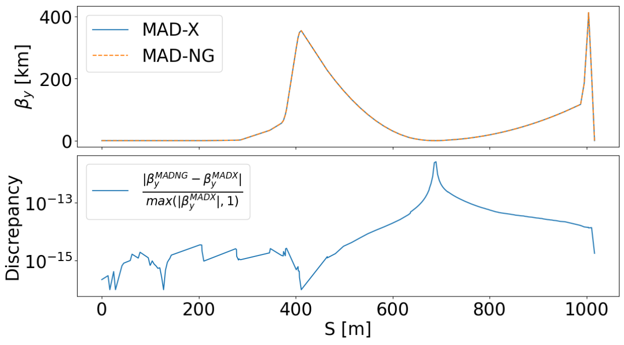


Figure 2: Vertical β function along the FFS computed with MAD-X (blue) and MAD-NG (orange) (top). Discrepancy between the MAD-X and MAD-NG outputs (bottom).

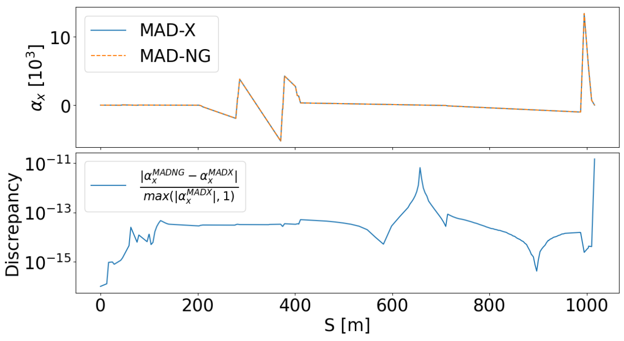


Figure 3: Horizontal α along the FFS computed with MAD-X (blue) and MAD-NG (orange) (top). Discrepancy between the MAD-X and MAD-NG outputs (bottom).

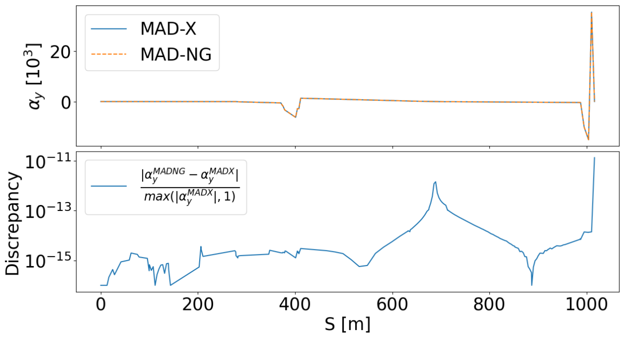


Figure 4: Vertical α along the FFS computed with MAD-X (blue) and MAD-NG (orange) (top). Discrepancy between the MAD-X and MAD-NG outputs (bottom).

The comparisons between MAD-X and MAD-NG for $\beta_{x,y}$, $\alpha_{x,y}$ and D_x in the CLIC FFS are shown in Figs. 1, 2, 3, 4 and 5. In order to have a stable comparison over a large dynamic range of variables values, the following

* enrico.manosperti@cern.ch

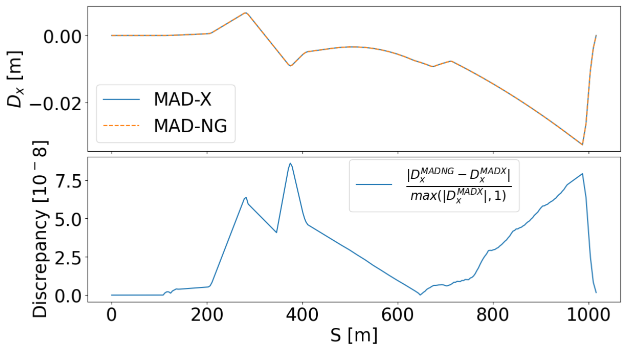


Figure 5: Horizontal dispersion along the FFS computed with MAD-X (blue) and MAD-NG (orange) (top). Discrepancy between the MAD-X and MAD-NG outputs (bottom).

formula is used for the discrepancy:

$$\text{Discrepancy} = \frac{|X^{\text{MADNG}} - X^{\text{MADX}}|}{max(|X^{\text{MADX}}|, 1)}\,, \tag{1}$$

where X is the value to be compared between the two codes. This formula behaves like an absolute error below one and a relative error above one. The *Twiss* command has been initialized with the values β_x=104.3 m, β_y=28.9 m, $\alpha_x = \alpha_y = 0$ and $D_x = 0$ m. The largest discrepancy between the two codes is reached for the dispersion which remains slightly below $\Delta D_x = 10^{-7}$ m. For β and α, the maximum discrepancy remains below 10^{-11}.

MATCHING

The *Match* [1,4] command is used for matching the optics parameters at any part of the machine. For this purpose, the strengths of the quadrupoles in the FFS (17 quadrupoles) will be varied, using the Jacobian method of the optimizer to reach the matching constraints in Table 1.

Table 1: Values at the IP Before Matching and the Targeted Values

Functions	Before matching	Targeted values
β_x [mm]	8.5	9
β_y [mm]	0.17	0.16
α_x	$6{\cdot}10^{-6}$	0
α_y	$22{\cdot}10^{-6}$	0
D_x [m]	$-1.4{\cdot}10^{-7}$	0

The optimizer will attempt to decrease the penalty function, which is a weighted mean square error estimator of the matched constraints calculated as follows:

$$P = \sqrt{\frac{\sum_i w_i^2 (x_i - c_i)^2}{\sum_i w_i^2}}, \tag{2}$$

where x_i and c_i are respectively the values and targets of the constraints, and the w_i are the weights associated with the constraints. Figure 6 shows the penalty function evolution during the optimization process for MAD-X and MAD-NG. The plot tracing the calls in MAD-NG shows the details of the optimization steps, starting with the computation of the Jacobian by finite differences in the variables (small plateaus of dots) and its application to the first new step (upper line), followed by a line-search bisection (single dots stepping down) attempting to decrease the penalty function, and where the maximum number of attempts is controlled by the *bisec* option (here *bisec*=3). Thus, the number of calls for each stage, as shown on the x-axis, is as follows:

$$N_i = \underbrace{\text{number of variables} + 1}_{\text{Jacobian evaluation}} + \underbrace{N_{\text{best bisec}}}_{\text{line bisection}}\,. \tag{3}$$

Unlike MAD-NG, MAD-X only considers one call for each N_i with no apparent jump on its slowly decreasing penalty function. The reason for this is that MAD-X seems to vary the variables in small steps, whereas MAD-NG looks for a global solution with larger initial steps before trying line bisection. Therefore, even if the MAD-NG penalty function increases initially, it converges to a better solution in this particular example.

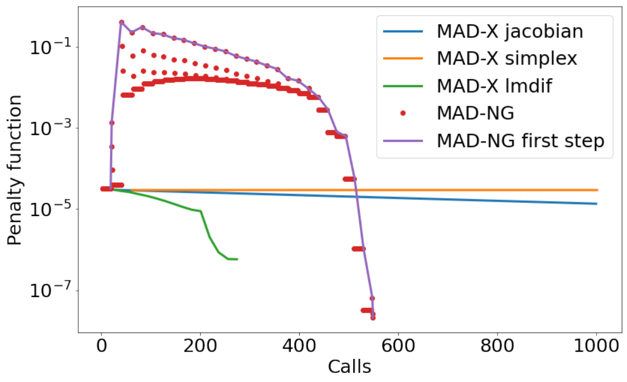


Figure 6: Penalty functions evolution versus the number of iterations.

SIMPLEX and LMDIF are other matching methods available in MAD-X. The SIMPLEX method is slower than the JACOBIAN method, but the LMDIF method [11] is the fastest, see Fig. 6). MAD-NG still reaches the best result of all as summarised in Table 2.

PERFORMANCE

The Polymorphic Tracking Code [12] (PTC) is a Fortran 90 library included in MAD-X that provides symplectic integration of the equations of motion through all accelerator elements. Both MAD-NG and PTC are polymorphic tracking codes based on the principle of tracking differential maps, with the main advantage of providing to users with many useful tools to manipulate these maps made out of TPSAs.

Execution Time

The running time of the MADX-PTC and MAD-NG *Track* commands were taken into account as well as the variation

Table 2: MAD-X and MAD-NG Values at the IP After Matching the Functions

Functions	MAD-X JACOBIAN	MAD-X SIMPLEX	MAD-X LMDIF	MAD-NG
β_x [mm]	8.8	8.5	9	9
β_y [mm]	0.17	0.17	0.17	0.16
α_x	$3{\cdot}10^{-6}$	$1.83{\cdot}10^{-7}$	$-4.3{\cdot}10^{-7}$	$6{\cdot}10^{-11}$
α_y	$-2{\cdot}10^{-6}$	$1.52{\cdot}10^{-7}$	$-1.1{\cdot}10^{-7}$	$8{\cdot}10^{-10}$
D_x [m]	$4{\cdot}10^{-12}$	$-1.45{\cdot}10^{-7}$	$4.9{\cdot}10^{-7}$	$-5{\cdot}10^{-9}$

of the map order, as shown in Fig. 7. In PTC it is possible to select the kind of Hamiltonian, exact or expanded [13], to be used for the elements maps through the *exact* option, i.e. *exact* = *true* (exact) or *exact* = *false* (expanded). Table 3

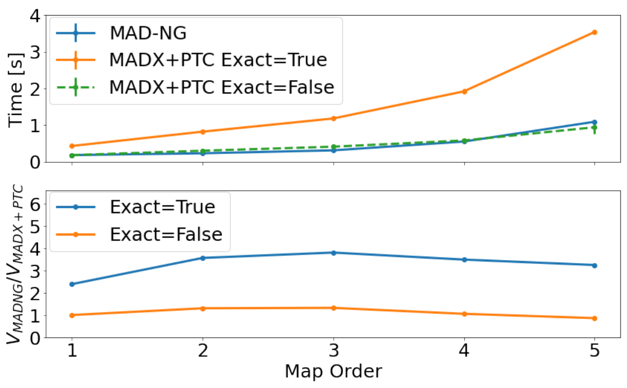
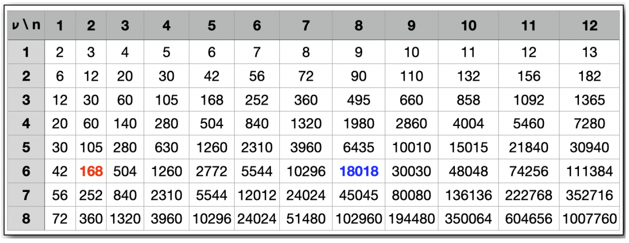


Figure 7: Execution time of the *Track* commands as a function of the map order (top). Ratio between MAD-NG and MADX-PTC execution speed (bottom).

shows the number of coefficients in the differential map, i.e. its size in memory, as a function of the number of variables, e.g. 6 variables in our case, and the order of its TPSAs, which both contribute to the calculation time required by the CPU to perform elementary arithmetic operations like additions (linear in size) and multiplications (quadratic in size). For this reason as well as cache size consideration, higher orders than the latter were not included in this benchmark. From Fig. 7, MAD-NG performs the map computation between two and four times faster than MADX-PTC for *exact=true* and at the same speed for *exact=false*.

Table 3: The Number of Coefficients in the TPSAs of a Differential Map With ν Variables at Order n is $\nu\binom{n+\nu}{\nu} = \frac{(n+\nu)!}{n!(\nu-1)!}$ [4]



ν \ n	1	2	3	4	5	6	7	8	9	10	11	12
1	2	3	4	5	6	7	8	9	10	11	12	13
2	6	12	20	30	42	56	72	90	110	132	156	182
3	12	30	60	105	168	252	360	495	660	858	1092	1365
4	20	60	140	280	504	840	1320	1980	2860	4004	5460	7280
5	30	105	280	630	1260	2310	3960	6435	10010	15015	21840	30940
6	42	168	504	1260	2772	5544	10296	18018	30030	48048	74256	111384
7	56	252	840	2310	5544	12012	24024	45045	80080	136136	222768	352716
8	72	360	1320	3960	10296	24024	51480	102960	194480	350064	604656	1007760

Results Accuracy

The maps generated by MAD-NG and MADX-PTC are used as input by Mapclass [14–17] to compute the higher-order beam size at the Interaction Point. As shown in Fig. 8, the beam size computed from the maps generated by the two codes, with both *exact* = *false* and *exact* = *true*, are the same within a tolerance below 10^{-2} nm.

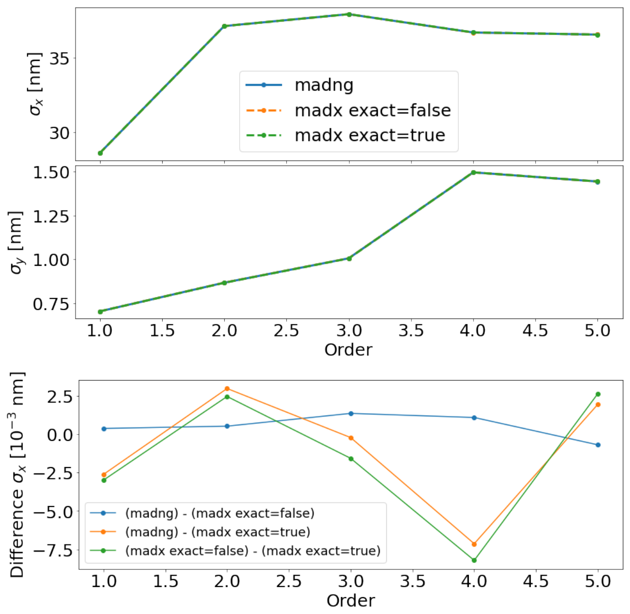


Figure 8: Beam size delivered by Mapclass with MAD-NG and MADX-PTC maps as input, computed here with a rms relative momentum spread of $\Delta p/p = 0.3\%$.

CONCLUSIONS

The comparison between MAD-X and the new code MAD-NG was carried out for the three main commands offered by these codes, namely *track*, *twiss*, and *match*. The evolution of the penalty functions and the delivered results obtained for different optimization methods have been analyzed to compare the overall performance of these codes. For the particular example shown here MAD-NG achieved a lower penalty function. Finally, the speed performance and the accuracy of the maps have been cross-checked between these codes for different setups showing better runtime performances for MAD-NG when considering the best available physics model *exact=true*.

REFERENCES

[1] MAD-X Home Page, `http://madx.web.cern.ch/madx`

[2] MAD-X Source Repository, `https://github.com/MethodicalAcceleratorDesign/MAD-X/`.

[3] W. Herr and F. Schmidt, "A MAD-X Primer", CERN-AB-2004-027-ABP, 2006, `https://cds.cern.ch/record/744163`, 10.5170/CERN-2006-002.505

[4] L. Deniau, *MAD-NG's Reference Manual*, `https://cern.ch/mad/releases/madng/html`

[5] L. Deniau, MAD-NG Source Repository, `https://github.com/MethodicalAcceleratorDesign/MAD`

[6] Lua Home Page, `https://www.lua.org`

[7] M. Pall, "The LuaJIT Project", `https://luajit.org`

[8] L. Deniau and C. I. Tomoiaga, "Generalised Truncated Power Series Algebra for Fast Particle Accelerator Transport Maps", in *Proc. IPAC'15*, Richmond, VA, USA, May 2015, pp. 374–377. `doi:10.18429/JACoW-IPAC2015-MOPJE039`

[9] M. Aicheler *et al.*, "A Multi-TeV Linear Collider Based on CLIC Technology: CLIC Conceptual Design Report", CERN, Geneva, Switzerland, Rep. CERN-2012-007, 2012. `doi:10.5170/CERN-2012-007`

[10] E. Manosperti, R. Tomás and A. Pastushenko *Design of CLIC Beam Delivery System at 7 TeV*, presented at IPAC'23, Venice, Italy, May 2023, paper MOPL113, this conference.

[11] F. James, *MINUIT: Function Minimization and Error Analysis Reference Manual*, 1998, `https://cds.cern.ch/record/2296388/files/minuit.pdf`

[12] E. Forest, F. Schmidt, and E. McIntosh, "Introduction to the Polymorphic Tracking Code: Fibre bundles, polymorphic Taylor types and "Exact tracking"", CERN, Geneva, Switzerland, CERN-SL-2002-044-AP, 2001. `http://cds.cern.ch/record/573082`

[13] E. Forest, "From Tracking Code to Analysis: Generalised Courant-Snyder Theory for Any Accelerator Model", Tokyo, Japan: Springer, 2016. `doi:10.1007/978-4-431-55803-3`

[14] *MAPCLASS*, `https://twiki.cern.ch/twiki/bin/view/ABPComputing/MapClass`

[15] R. Tomás, "Nonlinear optimization of beam lines", Phys. Rev. Spec. Top. Accel. Beams, vol. 9, 2006. `doi:10.1103/PhysRevSTAB.9.081001`

[16] R. Tomás, "MAPCLASS: a code to optimize high order aberrations", CERN, Geneva, Switzerland, AB-Note-2006-017, CERN-AB-Note-2006-017, 2006. `https://cds.cern.ch/record/944769`

[17] D. Martinez, A. Rosam, R. Tomás, and R. De Maria, "MAPCLASS2: a code to aid the optimisation of lattice design", CERN, Geneva, Switzerland, CERN-ATS-Note-2012-087, 2012. `https://cds.cern.ch/record/1491228`