

ADPO: Automatic-Differentiation-Assisted Parametric Optimization

Rong Chen, Mark Sale, Alex Mazur, Michael Tomashevskiy, Shuhua Hu, James Craig, Mike Dunlavey, Robert Leary, Keith Nieforth

Certara Inc., USA, 4 Radnor Corporate Center, Suite 350
Radnor, PA 19087, USA.

Background and Objective

- Automatic differentiation (AD) is a critical technique in modern machine learning for computing derivatives accurately (gradients obtained by AD are exact to machine precision) and efficiently [1]. However, the traditional method for gradient calculation in pharmacometrics (PMX) software is finite difference (FD). FD requires a nontrivial step size to calculate the approximated gradients and suffers from truncation errors and round-off errors.
- First order conditional estimation (FOCE) algorithm [2] – the most frequently used NLME (nonlinear mixed-effects) parameter estimation algorithm in PMX, heavily relies on gradient calculations as it uses the gradient-based Broyden–Fletcher–Goldfarb–Shanno (BFGS) quasi-Newton optimization method [3]. Traditionally, FD is used to calculate the gradients in BFGS.
- Our objective was to significantly improve the computational efficiency and maintain the accuracy of parameter estimation in Phoenix NLME’s [4] FOCE ELS algorithm by replacing traditional FD-based gradient calculations with AD.

Methods

- In FOCE ELS, the optimal population parameters estimates: fixed effects parameter vector θ , covariance matrix of random effects Ω , and σ which represent standard deviation(s) of residual error, are obtained via two levels of BFGS optimizations, i.e., the outer loop and the inner loop.
 - Outer loop: the optimal population parameters estimates θ , Ω and σ are obtained via iterations of BFGS algorithm by minimizing the negative logarithm of the overall data likelihood (-LL). Inside each BFGS iteration, to evaluate -LL, each subject’s optimal joint log-likelihood needs to be calculated which is done by the inner loop. The BFGS iterations continue, until -LL converges to a (local) minimum.
 - Inner loop: given the current estimates for θ , Ω and σ , each subject’s optimal joint log-likelihood is calculated from the corresponding optimal empirical Bayesian estimates (EBEs) of the random effect vector η , which is obtained via iterations of BFGS algorithm by minimizing the subject’s negative joint log-likelihood.
- Inner loop is embedded in the outer loop, so the improvement should begin from inner loop. We further identified the performance bottleneck of inner loop is the FD calculations in the BFGS algorithm.
- To overcome the bottleneck, we replace the FD calculations in the inner loop with AD via the dual numbers approach, which calculates the gradients accurately and is computationally efficient (the function value and its gradients are obtained simultaneously). We call the AD-based optimization as “automatic-differentiation-assisted parametric optimization” (ADPO).

Figure 1: ADPO as the “Fast Optimization” option in Phoenix 8.6.

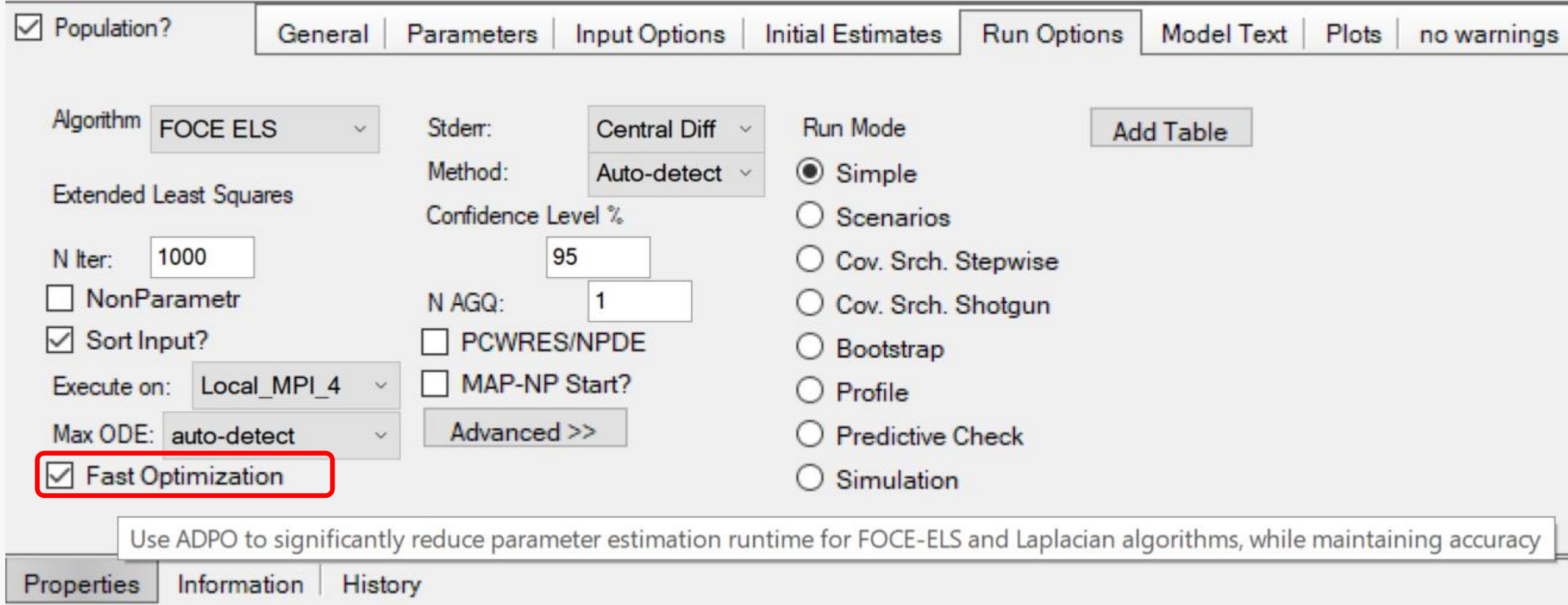
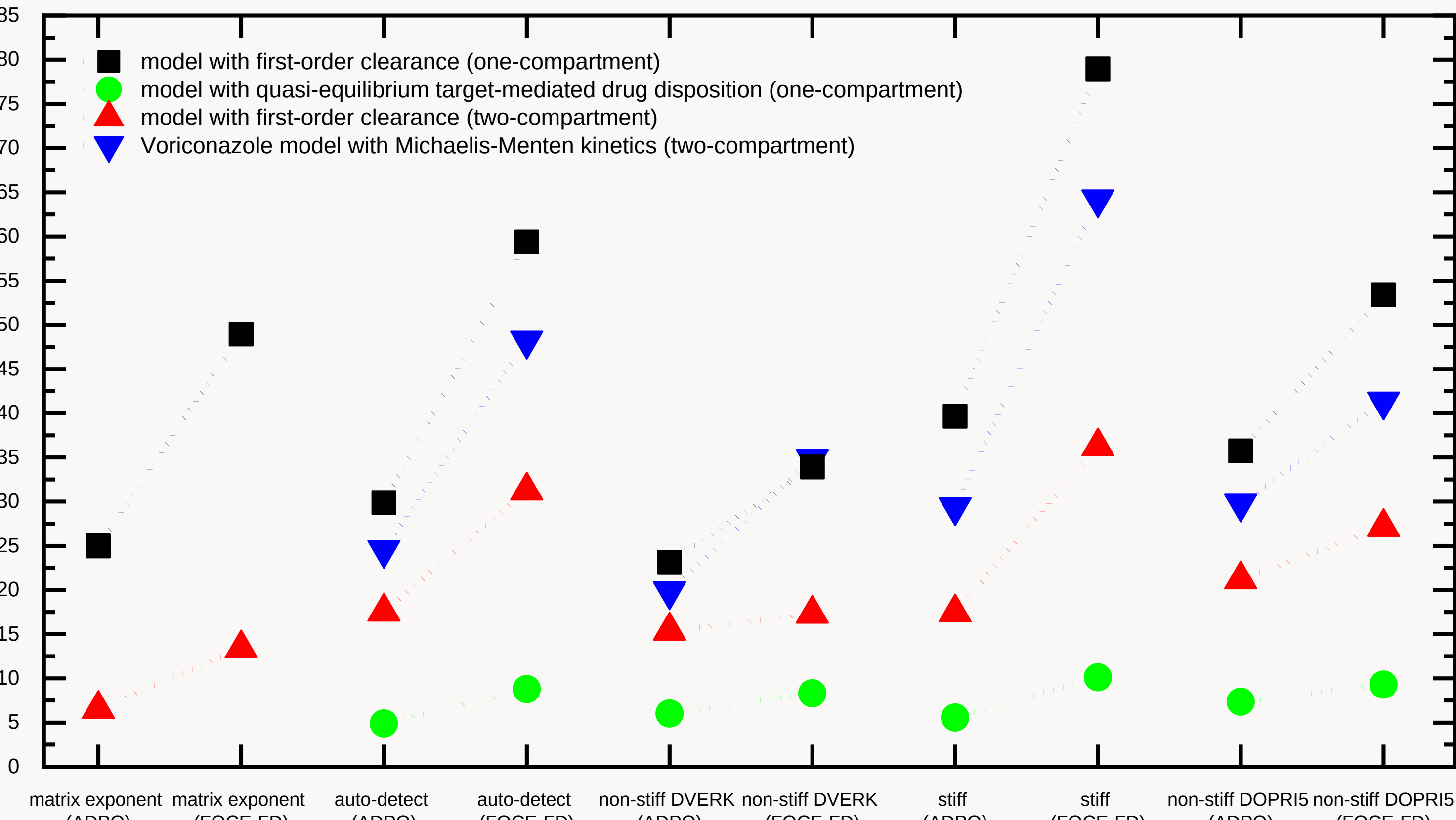


Figure 2: Performance of ODE solvers with and without ADPO.



Methods (Continued)

- In this poster, all the results are for FOCE ELS. When ADPO is enabled for FOCE ELS, the algorithm is marked as ADPO. The traditional FOCE ELS using FD, is marked as FOCE-FD.

Results

- ADPO is applicable to both FOCE ELS algorithm and Laplacian algorithms, since they share the same inner loop optimization procedure. In Phoenix NLME 8.6, ADPO can be enabled by simply checking the “Fast Optimization” box as shown in Fig. 1, for the corresponding algorithm.
- Most of the realistic models contain sets of ordinary differential equations (ODEs). The speed of the parameter estimation algorithms depend on the speed of the ODE solvers. Therefore, the speed of ADPO can be measured by the performance of the ODE solvers. In Fig. 2, for four different models, we measure the speed of the five ODE solvers (two of models contain non-linear ODEs, so “matrix exponent” solver cannot be used) available in Phoenix NLME with and without ADPO.
 - On the x-axis, the name of an ODE solver followed by “(FOCE-FD)” is the traditional one used in FOCE-FD, while the one ending with “(ADPO)” are the AD version of an ODE solver (all the calculations inside the solver has been promoted to dual number calculations).
 - On the y-axis is the time cost (measured in microsecond) for a solver to solve the ODEs in the model. For ADPO solvers, it is the “equivalent” time cost (ADPO solve the ODEs and the corresponding gradients simultaneously, while FOCE-FD solvers need to solve the ODEs n+1 times to get the solution of the ODEs and the gradients, where n is the number of random effects parameters in η), therefore, the time cost of ADPO solvers divided by n+1 is the “equivalent” time cost that is comparable with the FOCE-FD solvers’ time cost.
 - From Fig. 2, for each of the models, the time cost of ADPO solvers is significantly lower than the corresponding FOCE-FD solvers. It indicates ADPO is much faster than FOCE-FD.
- Table 1 shows the speed, accuracy and robustness comparisons of ADPO vs. FOCE-FD for a two-compartment model with first-order clearance. For each parameter, despite starting from an initial value that is different from the true parameters which are used to simulate the dataset, both FOCE-FD and ADPO converge to parameter estimates that are close to the true parameters and with reasonable standard error estimations, and both reach similar statistics such as LL, AIC and BIC. However, ADPO is about 20-50% faster in total run time and with much less backtracking in the inner loop BFGS optimizations. All these demonstrate that ADPO is much faster than FOCE-FD without compromising accuracy and robustness.

Conclusions

- Utilizing ADPO in the Phoenix NLME 8.6 engine provides significant speed improvements without sacrificing accuracy
- The performance advantage of ADPO over FOCE-FD becomes increasingly significant in the context of machine learning model selection where large numbers of models are executed.

Table 1: Speed, accuracy and robustness comparisons of ADPO vs. FOCE-FD for a two-compartment model with first-order clearance.

Parameter	Initial Value	True Value	Estimated parameter value (group by the available ODE solvers in Phoenix® NLME™) by FOCE ELS									
			matrix exponent		auto-detect		non-stiff DVERK		stiff		non-stiff DOPRI5	
			ADPO	FOCE-FD	ADPO	FOCE-FD	ADPO	FOCE-FD	ADPO	FOCE-FD	ADPO	FOCE-FD
$t_v V$	5.0	4.5	4.39 ±0.13	4.39 ±0.13	4.39 ±0.13	4.39 ±0.13	4.39 ±0.13	4.39 ±0.13	4.39 ±0.12	4.39 ±0.13	4.39 ±0.13	4.39 ±0.13
$t_v Cl$	1.0	1.6	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05	1.61 ±0.05
$t_v K_a$	1.0	0.8	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03	0.81 ±0.03
$t_v V_2$	10.0	10.5	10.28 ±0.15	10.28 ±0.15	10.27 ±0.15	10.28 ±0.15	10.28 ±0.15	10.28 ±0.15	10.26 ±0.15	10.28 ±0.15	10.28 ±0.15	10.28 ±0.15
$t_v Cl_2$	1.0	0.6	0.599 ±0.007	0.599 ±0.007	0.599 ±0.007	0.599 ±0.007	0.599 ±0.007	0.599 ±0.007	0.597 ±0.007	0.599 ±0.007	0.599 ±0.007	0.599 ±0.007
$t_v F_{logit}$	0.0	-1.6	-1.63 ±0.03	-1.63 ±0.03	-1.63 ±0.03	-1.63 ±0.03	-1.63 ±0.03	-1.63 ±0.03	-1.64 ±0.03	-1.63 ±0.03	-1.63 ±0.03	-1.63 ±0.03
σ	0.10	0.12	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002	0.120 ±0.002
ω_V	1.0	0.09	0.080 ±0.012	0.080 ±0.012	0.081 ±0.011	0.080 ±0.012	0.080 ±0.011	0.080 ±0.012	0.080 ±0.011	0.080 ±0.012	0.080 ±0.011	0.080 ±0.011
ω_{Cl}	1.0	0.09	0.083 ±0.011	0.082 ±0.011	0.083 ±0.010	0.082 ±0.011	0.083 ±0.010	0.082 ±0.011	0.082 ±0.010	0.083 ±0.011	0.083 ±0.011	0.083 ±0.011
ω_{K_a}	1.0	0.09	0.10 ±0.02	0.10 ±0.02	0.100 ±0.017	0.100 ±0.018	0.10 ±0.02	0.10 ±0.02	0.102 ±0.017	0.10 ±0.02	0.10 ±0.02	0.10 ±0.02
$\omega_{F_{logit}}$	1.0	0.09	0.086 ±0.013	0.086 ±0.014	0.086 ±0.014	0.086 ±0.013	0.086 ±0.013	0.086 ±0.013	0.086 ±0.011	0.086 ±0.014	0.086 ±0.013	0.086 ±0.013
Log-likelihood (LL)			-2851.460	-2851.461	-2851.461	-2851.468	-2851.460	-2851.461	-2851.481	-2851.462	-2851.461	-2851.461
Akaike Information Criterion (AIC)			5724.921	5724.922	5724.922	5724.936	5724.921	5724.922	5724.962	5724.925	5724.921	5724.922
Bayesian Information Criterion (BIC)			5784.077	5784.077	5784.078	5784.091	5784.076	5784.077	5784.118	5784.081	5784.077	5784.077
Estimation run time (seconds)			5.828	11.578	23.156	46.992	14.219	22.141	24.453	60.516	20.906	25.125
Standard error run time (seconds)			1.703	3.641	5.875	11.109	4.016	5.206	6.594	11.906	5.891	7.625
Total run time (seconds)			7.531	15.219	29.031	58.101	18.233	27.407	31.047	72.422	26.797	32.775
Total # of outer loop BFGS iterations			70	65	66	74	71	75	63	80	71	71
Total # of outer loop backtracks			6	20	21	14	31	24	17	29	19	8
Total # of inner loop BFGS iterations			230178	226787	281119	318645	236773	315861	312136	359044	250670	238333
Total # of inner loop backtracks			1231	3796	140107	266393	1254	4236	105231	203876	8825	16801

References

- Baydin, A.G., et al., Automatic differentiation in machine learning: a survey. Journal of machine learning research, 2018. 18(153): p. 1-43.
- Wang, Y., Derivation of various NONMEM estimation methods. Journal of Pharmacokinetics and pharmacodynamics, 2007. 34(5): p. 575-593.
- Dennis Jr, J.E. and R.B. Schnabel, Numerical methods for unconstrained optimization and nonlinear equations. 1996: SIAM.Certara. Phoenix NLME. 2025; Available from: <https://www.certara.com/software/pkpd-modeling-and-simulation/phoenix-nlme/>.



Want to learn more?
<< Scan Here