OPTIMIZATION OF SURFACE KINETICS SCHEMES VIA MACHINE LEARNING METHODS

José Afonso [*], **Vasco Guerra**[1], and **Pedro Viegas**[1]

[1]Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico, Av. Rovisco Pais 1, Lisbon, Portugal

ABSTRACT

The accurate modeling of plasma-surface interactions is often hindered by uncertainties in experimental measurements of surface recombination kinetics. This work addresses this challenge by framing the problem within a probabilistic framework. We derive a principled objective loss function directly from this framework, which is then used to optimize a surface kinetics scheme. Furthermore, we introduce an optimization algorithm specifically tailored to our simulator, designed to efficiently navigate the sloppy nature inherent in this class of problems. Comparative analysis demonstrates that our proposed algorithm consistently outperforms baseline optimization methods, offering a more robust and reliable approach for determining surface kinetics. The implementation is presented in https://github.com/joseAf28/PlasmaDM.

***K*eywords** Surface Kinetics schemes · Optimization Methods · Model-free methods · Machine Learning

With four knobs, we conjure an elephant; with five, we make it wiggle its trunk. However, one parameter in one form can blossom into many in another, and it's in those hidden, effective dimensions that the physics resides.

1 Introduction

The study of plasma-surface interactions is essential to fields ranging from microelectronics fabrication to fusion energy. These phenomena are typically described by deterministic models based on systems of differential equations, which depend on numerous parameters within a surface kinetics scheme [1, 2, 3]. While some of these parameters are constrained by physical theory or direct measurement, many others, such as the energy barriers for chemical reactions, remain poorly defined in the literature [2].

Optimizing plasma-surface kinetics is therefore of great importance, despite being hindered by two main obstacles. First, the simulations themselves could have an expensive computational cost, limiting the feasibility of exhaustive (naive) parameter searches. Second, the inclusion and mitigation of the parameter uncertainty estimation. This work addresses this challenge by employing a data-driven framework to calibrate and refine the physical models against experimental observations.

This calibration task requires navigating a complex optimization landscape, which ill-conditioned and non-convex, featuring strong parameter correlations and numerous local minima. To tackle this, we systematically investigate a range of optimization strategies. We begin by establishing a baseline using robust global search algorithms, such as Differential Evolution (DE) [4], and Local Based Methods [5]. Recognizing the limitations of these approaches, we then introduce our primary contribution: a hierarchical optimization strategy. This method decomposes the problem by leveraging its inherent physical structure to achieve superior results with a fraction of the computational cost.

The successful optimization and refinement of this kinetics scheme provide two key benefits. First, it yields valuable insights into the relationships between different parameters, enhancing the predictive capabilities of the physical models. Second, it helps delineate the models' limitations, offering new perspectives on regimes not well-explained by current

[*]Corresponding author: josefafonso@tecnico.ulisboa.pt

formulations. Furthermore, through an analysis of the loss manifold, this study examines why reliable predictions can often be made despite significant parameter uncertainty.

This work is organized as follows: Section 2 presents the theoretical background and the derivation of our objective function. Section 3 details the optimization methodologies, from established global methods to our proposed hierarchical algorithm. Section 4 presents the results of applying these methods to O_2 and CO_2 kinetic schemes. Finally, Section 5 discusses our findings and presents the conclusions.

2 Theoretical Backgroud

2.1 General (Probabilistic) Formulation

In this section, we present a derivation of the general probabilistic formulation that we will consider. We used this formulation motivated by the fact that our experimental conditions and measured observables are inherently noisy, despite having a deterministic physical simulator [2]. Additionally, it creates a transparent framework for uncertainty quantification and model comparison, clarifying the approximations used to derive the objective functions. Finally, this formulation enhances our understanding of the variable dependency within our physical system.

We can think of the experimental dataset as:

$$\mathcal{D} = \{(x_i^{\text{exp}}, \gamma_i^{\text{exp}})\}_{i=0}^{N} \sim p^*(x^{\text{exp}}, \gamma^{\text{exp}}) \tag{1}$$

where $p^*(x^{\text{exp}}, \gamma^{\text{exp}})$ is the unknown *ground-truth* joint over input experimental x^{exp} and the measured recombination probabilities γ^{exp}.

Our physical simulator, parametrized by θ, defines a joint distribution $p_\theta(x^{\text{exp}}, \gamma)$. From this, we generate a synthetic dataset.

$$\mathcal{D}_\theta = \{(x_i^{\text{exp}}, \gamma_i)\}_{i=1}^{N} \sim p_\theta(x^{\text{exp}}, \gamma) \tag{2}$$

In the most general sense, we can define our optimization problem as:

$$\theta^* = \arg \min_{\theta \in \Theta_{\text{phys}}} D(p^* \,||\, p_\theta) \tag{3}$$

where $D(p^*||p_\theta)$ corresponds to a divergence (or distance) between probability distributions. Moreover, the problem consists of finding the point, θ, in the physically allowable parameter space that makes the simulator's distribution, p_θ, as close as possible to the experimental one, p^*.

2.2 Graphical Model for the Simulator Joint Distribution

Now, we make explicit assumptions behind the simulator joint distribution, $p_\theta(x^{\text{exp}}, \gamma)$. We start by introducing a latent *true* condition x^* and treat both the observed inputs x^{exp} and outputs γ as random variables conditioned on this latent quantity and on the model hyperparameters θ.

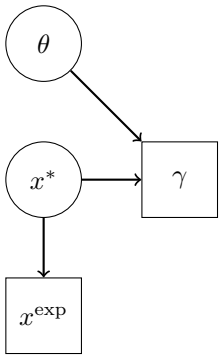

Figure 1: Graphical model for the probabilistic formulation.

In our formulation, the nodes have the following meaning:

- θ (latent) : the set of hyperparameters of our model (energy barriers, steric factors, ...),

- x^* (latent) : the *true* (unobserved) experimental conditions (e.g., wall temperature, gas species densities, ...),
- x^{exp} (observed) : the noisy measurement of condition x^* that the instruments actually record,
- γ (observed): the prediction of the recombination probability,

while the edges represent the conditional dependencies.

Concretely, for each point i from the dataset, we assume Gaussian noise models:

$$x_i^{\text{exp}} \mid x_i^* \sim \mathcal{N}\big(x_i^*,\, \Sigma_{x,i}\big), \tag{4}$$

$$\gamma_i \mid x_i^*,\, \theta \sim \mathcal{N}\big(\gamma(x_i^*, \theta),\, \sigma_{y,i}^2\big), \tag{5}$$

where $\mathcal{N}(\mu,\, V)$ denotes a Gaussian with mean μ and covariance V, $\Sigma_{x,i}$ is the assumed covariance of the input measurement noise for point i, $\sigma_{y,i}^2$ is the variance of the output measurement noise for point i, and $\gamma(x_i^*, \theta)$ is the model's prediction of the recombination probability at true condition x_i^*.

Because each experiment $(x_i^{\text{exp}}, \gamma_i, x_i^*)$ is drawn independently from the same joint distribution, and our priors over the x_i^* and over θ are uniform (and independent of one another), the full joint factorizes as:

$$p\big(\{x_i^{\text{exp}}, \gamma_i, x_i^*\}_{i=1}^M, \theta\big) = p(\theta) \times \prod_{i=1}^{M} p(x_i^*) \times p\big(x_i^{\text{exp}} \mid x_i^*\big) \times p\big(\gamma_i \mid x_i^*, \theta\big). \tag{6}$$

Since x_i^* is a latent variable and hence it is never observed, the predicted observed-data pair $(x_i^{\text{exp}}, \gamma_i)$ likelihood is

$$p_\theta(x_i^{\text{exp}}, \gamma_i) = p(x_i^{\text{exp}}, \gamma_i|\theta) = \int p(x_i^{\text{exp}}, \gamma_i, x_i^* \mid \theta)\, dx_i^* = \int p(x_i^{\text{exp}}|x_i^*)\, p(\gamma_i \mid x_i^*, \theta)\, p(x_i^*)\, dx_i^* \tag{7}$$

By plugging the Gaussian models presented in Eqs.[4-5] and assuming the uniform prior on x_i^*, we get

$$p(x_i^{\text{exp}}, \gamma_i|\theta) = \int dx_i^*\, \mathcal{N}(x_i^{\text{exp}}|x_i^*, \Sigma_{x,i}) \times \mathcal{N}(\gamma_i|\gamma(x_i^*, \theta), \sigma_{y,i}^2) \tag{8}$$

2.3 Physical Simulator Formulation

Now we address the physical simulator surface theoretical formulation. The system's general formulation is split into two parts. The first one involves the calculation of the average surface density species by solving a system of differential equations, which is generally given by:

$$\frac{d}{dt}\vec{y}(t) = \vec{F}(\vec{x}, \vec{y}(t); \theta), \;\; \vec{y}(t_0) = \vec{y}_0 \tag{9}$$

where $\vec{x}$ is the vector of the input experimental conditions (e.g., wall temperature, gas temperature, ...) , $\vec{y}$ are the average species concentrations, θ the set of parameters that govern the surface kinetics (e.g. energy activation barrier, desorption frequencies, ...) and $F(\cdot)$ is the system of chemical equations that model the surface, and is obtained from the kinetics scheme. A rigorous explanation of its derivation is presented in [1, 2, 3].

Our work will focus on the $\mathrm{O}_2 - \mathrm{CO}_2$ surface kinetic scheme. The surface kinetic scheme follows exactly the one presented in [6] and its physical derivation and motivation are widely established in [1, 2, 3]. In this specific case, $\vec{y}$ is given by:

$$\vec{y}^T = (\mathrm{F}_V \;\; \mathrm{O}_F \;\; \mathrm{CO}_F \;\; \mathrm{O}_{2\,F} \;\; \mathrm{S}_V \;\; \mathrm{S}_V^* \;\; \mathrm{O}_S \;\; \mathrm{O}_S^* \;\; \mathrm{CO}_S \;\; \mathrm{CO}_S^*) \tag{10}$$

where the species that have * correspond to metastable species. All species containing F occupy physisorption sites, while those involving S occupy chemisorption sites. The primary distinction between physisorption and chemisorption lies in the strength of the interactions. In physisorption, particles bond to the surface through *van der Waals* forces, while in chemisorption, chemical bonds are formed [1].

We seek, for each fixed experimental condition $\vec{x}$ and kinetic parameter set θ, the nonnegative steady-state:

$$\vec{y}^* = \vec{y}^*(\vec{x}, \theta), \tag{11}$$

characterized by $\vec{F}(\vec{x}, \vec{y}^*; \theta) = \vec{0}$, assumed to be unique in the physically admissible domain $\{\vec{y} \geq 0\}$ and *asymptotically stable* ($\lim_{t\to\infty} \vec{y}(t) = \vec{y}^*$), which implies $\vec{y}^*$ does not depend on the choise of initial conditions $\vec{y}_0$.

The second part involves computing the macroscopic observables, the recombination probability, γ, the quantity that can be measured. It is generally given by:

$$\gamma = \hat{T}(\vec{x}, \vec{y}^*(\vec{x}, \theta), \theta), \tag{12}$$

where $\hat{T}(\cdot)$ corresponds to an operator that, by selecting the appropriate reactions from the surface kinetics scheme, projects the computed steady-state chemical concentrations, $\vec{y}^*$, into the scalar and observable quantity γ. More details and the derivation of this observable are widely explained in [1].

3 Methodology

3.1 Objective Function Derivation

Having presented the general formulation in Section 2 , we now derive the objective functions used for solving the general optimization problem presented in Eq.[3]. We start by picking a divergence measure for evaluating Eq.[3]. We will use the *Kullback-Leibler* divergence [7, 8] , D_{KL}, given its widely used in Optimization and Machine Learning problems. As a result, we obtain:

$$\theta^* = \arg\min_{\theta\in\Theta_{\mathrm{phys}}} D_{\mathrm{KL}}(p^*||p_\theta) \tag{13}$$

$$= \arg\min_{\theta\in\Theta_{\mathrm{phys}}} \int dx\, d\gamma\; p^*(x,\gamma)\log\frac{p^*(x,\gamma)}{p_\theta(x,\gamma)} \tag{14}$$

By neglecting the terms that are independent w.r.t θ, we have that:

$$\theta^* = \arg\max_{\theta\in\Theta_{\mathrm{phys}}} \int dx\, d\gamma\; p^*(x,\gamma)\log p_\theta(x,\gamma) \tag{15}$$

$$= \arg\max_{\theta\in\Theta_{\mathrm{phys}}} \mathbb{E}_{(x,\gamma)\sim p^*}\left[\log p_\theta(x,\gamma)\right] \tag{16}$$

Since we do not know p^* in closed form, only a dataset $\mathcal{D}_\theta$ drawn from it, as presented in Eq.[1], we replace the true expectation by the empirical one, which allows us to obtain:

$$\theta^* \approx \arg\max_{\theta\in\Theta_{\mathrm{phys}}} \mathcal{L}(\theta), \tag{17}$$

with

$$\mathcal{L}(\theta) = \mathbb{E}_{(x^{\exp},\gamma^{\exp})\sim\mathcal{D}}\left[\log p_\theta(x,\gamma)\right] = \sum_{i=1}^{M}\log p(x_i^{\exp},\gamma_i^{\exp}|\theta), \tag{18}$$

where we drop the $1/M$ since it does not affect the $\arg\max$. We see that opting for the KL-divergence choice, we end up obtaining as the objective function the same that we obtain from *log-likelihood* formalism.

By formulating our problem in the context of this probabilistic framework, we can accurately address input uncertainties within the model, instead of assuming that experimental conditions are error-free. This approach allows us to obtain well-founded error bars on both predictions and parameter estimates. Additionally, we can utilize techniques such as Gaussian approximation and variational inference [9] to fit the model. Furthermore, it provides a way of deriving optimization objectives in a more rigorous way.

3.2 Gaussian Approximation

Our goal is to evaluate (or optimize) the observed-data (marginal) likelihood

$$p\big(x_i^{\exp},\gamma_i^{\exp}\mid\theta\big) \;=\; \int p(\gamma_i^{\exp}\mid x_i^*,\theta)\,p(x_i^{\exp}\mid x_i^*)\;dx_i^*, \tag{19}$$

which is intractable as $\gamma(x^*,\theta)$ is nonlinear. We therefore approximate this integral by using a first-order Taylor (Laplace) expansion of the forward model around the observed input $x_i^{\exp}$.

Assuming that $\gamma(x,\theta)$ is smooth in x, we do the first-order Taylor expansion of γ around x_i^{exp}:

$$\gamma(x^*,\theta) \approx \gamma(x^{\exp},\theta) + J_x(\theta)(x^* - x^{\exp}), \quad J_x(\theta) = \nabla_x\gamma(x,\theta)\mid_{x^{\exp}} \tag{20}$$

Since $\delta x_i = x_i^* - x_i^{\exp} \sim \mathcal{N}(0,\Sigma_{x,i})$, and $\varepsilon_i \sim \mathcal{N}(0,1)$, we have

$$\gamma_i^{\exp} = \gamma(x^*,\theta) + \sigma_{y,i}\,\varepsilon \approx \gamma(x_i^{\exp},\theta) + J_{x,i}(\theta)\delta x_i + \sigma_{y,i}\,\varepsilon_i. \tag{21}$$

Which allows us to conclude that

$$\gamma_i^{\exp} \sim \mathcal{N}(\mu_i, V_i), \quad \text{with } \mu_i = \gamma(x_i^{\exp},\theta), \;\; V_i = \sigma_{y,i}^2 + J_{x,i}(\theta)\Sigma_{x,i}J_{x,i}^T(\theta). \tag{22}$$

After performing some algebraic manipulations, we obtain aside from constants w.r.t θ, the approximate *log-likelihood*:

$$\mathcal{L}_{\mathrm{approx}}(\theta) \approx -\frac{1}{2}\sum_{i=1}^{M}\left[\frac{(\gamma_i^{\exp}-\mu_i)^2}{V_i} + \log V_i\right]. \tag{23}$$

The full derivation is presented in Appendix 6.1.

In the limiting case where the input noise is negligible, $\Sigma_{x,i} \to \mathbf{0}$ (or $J_{x,i} \to 0$), the recombination probability measurement follows $\sigma_{y,i} = \alpha\gamma_i^{exp}$ and we neglect constants w.r.t θ (not important to find the best hyperparameters), we end up with the following objective loss:

$$\Phi(\theta) = -\frac{1}{2\alpha^2}\sum_{i=1}^{M}\left(\frac{\gamma_i^{\exp}-\mu_i}{\gamma_i^{\exp}}\right)^2, \tag{24}$$

which corresponds to a scale-invariant objective loss, which naturally down-weights points with larger observed γ.

3.3 γ Error Propagation

We now intend to estimate $p(\gamma|\mathcal{D},\theta)$, the full predictive distribution, which allows us, based on experimental uncertainties of our dataset and the hyperparameters θ, to quantify the error uncertainty of our model predictions.

Using the presented formalism developed through the graphical model [1] , and focusing on $(x_i^{\exp},\gamma_i^{\exp})$, a point of $\mathcal{D}$, we compute $p(\gamma \mid x_i^{\exp},\gamma_i^{\exp},\theta)$, which is given by:

$$p(\gamma_i|x_i^{\exp},\gamma_i^{\exp},\theta) = \int dx_i^*\; p(\gamma_i|x_i^*,\theta)\times p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta). \tag{25}$$

Since no closed form exists, as $\gamma(x,\theta)$ is nonlinear, we use Monte Carlo sampling [8]. Following the procedure, we start by drawing S samples from

$$x_i^{*(s)} \sim p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta), \quad s=1,\dots,S \tag{26}$$

For each sample, we compute the simulator output

$$\gamma_i^{(s)} \sim \mathcal{N}(\gamma(x_i^{*(s)},\theta),\sigma_{y,i}^2), \tag{27}$$

which enables us to obtain the empirical distribution, $\hat{p}(\gamma \mid \theta)$:

$$\hat{p}(\gamma_i|\theta,\mathcal{D}) = \frac{1}{S}\sum_{s=1}^{S}\delta_{\gamma_i^{(s)}}(\gamma). \tag{28}$$

To apply this procedure, we need to sample from $p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta)$. Although it is directly unknown, we can deduce its properties by utilizing the graphical model [1]. Hence, we obtain:

$$p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta) = \frac{p(x_i^*,x_i^{\exp},\gamma_i^{\exp}|\theta)}{p(x_i^{\exp},\gamma_i^{\exp}|\theta)} \propto p(x_i^*,x_i^{\exp},\gamma_i^{\exp}|\theta) = p(x_i^*)\times p(x_i^{\exp}|x_i^*)\times p(\gamma_i^{\exp}|x_i^*,\theta). \tag{29}$$

Using the result of Eq.[22] for estimating $p(\gamma_i^{\exp}|x_i^*,\theta)$, we have

$$p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta) \propto \exp -\frac{1}{2}\delta x^T\Sigma_{x,i}^{-1}\delta x - \frac{1}{2\sigma_{y,i}^2}(\gamma_i^{\exp}-\mu_i-J_{x,i}\delta x)^2, \tag{30}$$

where $\delta x = x^* - x^{\exp}$. By using the general expression of the square completion for δx, we have:

$$p(\delta x|\gamma^{\exp},\theta) \propto \exp -\frac{1}{2}(\delta x-m)^T C^{-1}(\delta x-m), \tag{31}$$

with

$$C = \left(\Sigma_x^{-1}+\frac{1}{\sigma_y^2}J_x^TJ_x\right)^{-1} \qquad \text{and} \qquad m_i = C_i\left(\frac{1}{\sigma_y^2}J_x^T(\gamma^{\exp}-\mu)\right). \tag{32}$$

Thus, we conclude that:

$$p(\delta x_i|\gamma_i^{\exp},\theta) \approx \mathcal{N}(m_i,C_i). \tag{33}$$

The full derivation of $p(\delta x \mid \gamma^{\exp},\theta)$ is presented in Appendix 6.2.

In this way, we approximate the samples drawn by $p(x_i^*|x_i^{\exp},\gamma_i^{\exp},\theta)$ as:

$$x_i^* \sim \mathcal{N}(x_i^{\exp}+m_i;C_i). \tag{34}$$

As a limiting case, where we assume that $J_{x,i}\to 0$, we have:

$$x_i^* \sim \mathcal{N}(x_i^{\exp},\Sigma_{x,i}). \tag{35}$$

This result implies that $\gamma(x,\theta)$ carries no information about x^* and the posterior on x^* reverts to the measurement $x^{\exp}$ and its covariance $\Sigma_{x,i}$.

3.4 Optimization Problem

Following the presentation and derivation of the general probabilistic formulation in Section [2.2], we now introduce the objective function that is to be optimized.

Given that our numerical physical simulator, LoKI-B+C [10], does not incorporate an automatic differentiation module, and to minimize additional computational overhead, we select an objective loss function based on the one outlined in Eq.[24]. Thus, we consider the following optimization problem:

$$\theta^* = \arg\min_{\theta \in \Theta} \tilde{\Phi}(\theta) \tag{36}$$

where $\Theta \subseteq \mathbb{R}^d$ is the domain of physically admissible parameter vectors, and $\tilde{\Phi} : \Theta \to \mathbb{R}$ is the objective loss to be minimized and it is given by:

$$\tilde{\Phi}(\theta) = \frac{1}{2} \sum_{i=1}^{M} r_i^2(\theta), \qquad r_i(\theta) = \frac{\gamma_i^{\exp} - \mu_i(\theta)}{\gamma_i^{\exp}}. \tag{37}$$

Among the various optimization techniques available, we can categorize them into two broad types: **model-free** and **model-based** methods. In this context, we choose to focus on the model-free approaches.

On the one hand, model-free optimization excels when the simulator we sample from is fast, efficient, parallelizable, and even non-smooth, because it treats the simulator as an *oracle* and never attempts to fit an explicit model of its response surface [11, 12]. In fact, we are in this regime: $\sim 10\,\mathrm{secs}$ per objective loss call - iteration. Thus, without a surrogate model to train or update, every CPU cycle is saved to evaluate new parameter sets, resulting in lower overhead and faster wall-clock convergence.

On the other hand, model-free optimizers do not assume continuity or differentiability, making them reliable in situations where the kinetic scheme experiences abrupt transitions or is poorly conditioned.

3.5 Optimization Algorithms

3.5.1 Global Derivative-Free Methods

As a baseline, we first apply global optimization methods designed to explore the entire parameter space without requiring gradient information. These *black-box* methods are important as they allow us to understand the overall landscape of the objective function.

Differential Evolution (DE)

First, we utilize Differential Evolution (DE), a robust, population-based stochastic algorithm [4]. DE evolves a population of parameter vectors by creating new candidates through a process of mutation (adding scaled difference vectors), crossover (mixing parent and mutant vectors), and selection [4, 13].

It has the benefit of allowing global exploration, which helps it escape local minima, and its inherent parallelizability, as each member of the population can be evaluated independently. However, it has two key limitations for our use case. First, its performance degrades significantly with the number of parameters (the *curse of dimensionality*). Second, it struggles with ill-conditioned problems, where the objective function is vastly more sensitive to some parameters than others, as shown in Fig.[2]. This is because DE typically uses isotropic mutation strategies that are not adapted to the local loss landscape [13].

Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

To complement the exploratory attributes of DE, we employ the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [14]. Unlike model-free methods, CMA-ES is effectively a model-based strategy that builds and refines a probabilistic model of the search landscape, represented by a multivariate normal distribution $\mathcal{N}(m, \sigma^2\, C)$. At each generation, the algorithm updates the distribution's mean m, step size σ, and covariance C based on the success of sampled candidate solutions.

The key strength of this approach is the adaptation of the covariance matrix, which enables the search distribution to learn the geometry of the loss landscape. It infers the problem's second-order structure by adapting its covariance matrix, which serves as an analogue of the inverse Hessian [14, 15]. However, for our case, this stochastic, sample-based process is different from leveraging explicit structural information, such as computing the Hessian matrix, and it requires a significant computational effort in each generation.

Moreover, it is susceptible to the initial step size hyperparameter, which must be balanced to avoid convergence to the nearest local minimum.

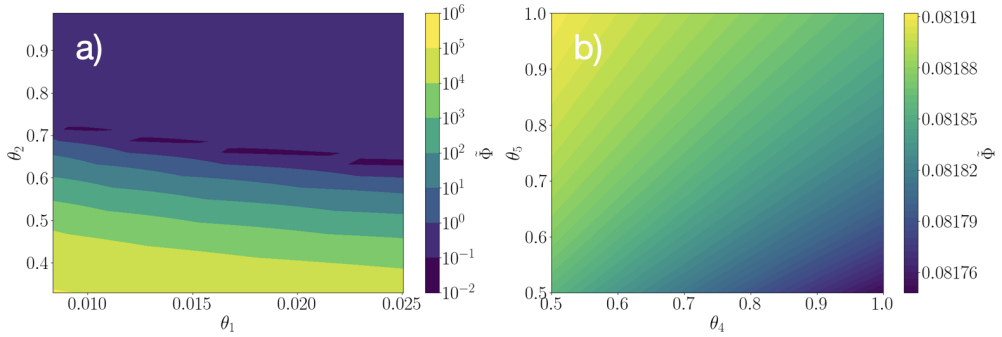

Figure 2: Contour plots of the objective loss $\tilde{\Phi}$ across pairs of hyperparameters. (a) Loss as a function of θ_1 and θ_2, showing a highly sensitive (steep) landscape, which means that tuning these parameters has a substantial impact on total error. The θ_1 and θ_2 correspond to parameters that tune the energy barrier of $O_F, O_{2,F}$ desorption frequency. (b) Loss as a function of θ_3 and θ_4, showing a nearly flat landscape. Changes in these parameters yield minimal variation in total error. These parameters correspond to sterical factors from the CO kinetics scheme.

3.5.2 Local Search Methods

While global optimization methods search the entire parameter space, we now turn to local optimization methods. These methods start from an initial guess and iteratively refine the solution by exploring its immediate neighborhood. We can distinguish between two main approaches: those that operate without gradient information and those that leverage it for a more guided search [5].

Gradient-Free Methods

The most direct local search methods work without requiring derivative information, treating the objective function $\tilde{\Phi}(\theta)$ as a *black-box*. Based on the principles of derivative-free optimization [5], the methods we explore include the Nelder-Mead method [16], which uses a simplex (a geometric shape of $n+1$ vertices in n dimensions) to explore the parameter space, and Powell's method [17], which performs sequential line searches along a set of optimized directions. While these methods are straightforward to use and can be effective on low-dimensional, smooth problems, their convergence can be slow. They often struggle to escape the sharp, narrow valleys of objective loss. They also become significantly less efficient in higher-dimensional problems [5].

Gradient-based Methods

Instead of treating $\tilde{\Phi}(\theta)$ as a pure *black box*, we utilize gradient information to guide a constrained local solver [18]. The gradient itself can be obtained in two ways. First, through finite differences, where the gradients are approximated numerically by independently perturbing each parameter θ_j. While straightforward, this method's computational cost scales linearly with the number of parameters, at $\mathcal{O}(n)$ per evaluation. Alternatively, Automatic Differentiation (AD) can compute the exact gradients via backpropagation, provided the simulator $\gamma(x,\theta)$ (Eq.[12]) and the objective function $\tilde{\Phi}$ are expressed in a supported framework like PyTorch [19]. For our case, this is given by:

$$\frac{\partial\tilde{\Phi}}{\partial\theta_j} = \sum_{i=1}^{M} \frac{\gamma_i(\theta) - \gamma_i^{exp}}{(\gamma_i^{exp})^2} \frac{\partial\gamma_i(\theta)}{\partial\theta_j}, \tag{38}$$

The remarkable advantage of AD is that the cost of computing the full gradient is roughly constant with respect to the number of parameters, $\mathcal{O}(1)$, making it efficient for high-dimensional problems. The full derivation of the analytic gradient is provided in Appendix [6.3].

With the gradient available, we can employ a constrained local solver like L-BFGS-B [20]. Since these solvers only find the nearest local minimum, the final solution is highly dependent on the starting point. To address this, we adopt a *multi-start* strategy as proposed by Ugray et al. [21], launching the optimizer from multiple random initial points θ_0 to increase the probability of discovering the global minimum.

This gradient-based local approach offers several clear benefits. It ensures fast local convergence, as taking descent steps guarantees a reduction in $\tilde{\Phi}$ at each iteration. The multi-start strategy is also highly parallelizable, and the use of AD allows the method to scale well to high-dimensional problems.

However, this approach has its drawbacks. The primary caveat is its dependence on the initial guess, as it only converges to a nearby stationary point. Furthermore, it can be quite inefficient in flat regions of the objective landscape, where

gradient-based progress is slow. The use of AD also introduces a significant implementation overhead, requiring the simulator to be rewritten in an AD-enabled library, which is not always feasible. Finally, the method can suffer from ill-conditioning if gradient components vary by several orders of magnitude, leading to oscillations or extremely slow convergence along stiff directions [22], as demonstrated in Fig.[3].

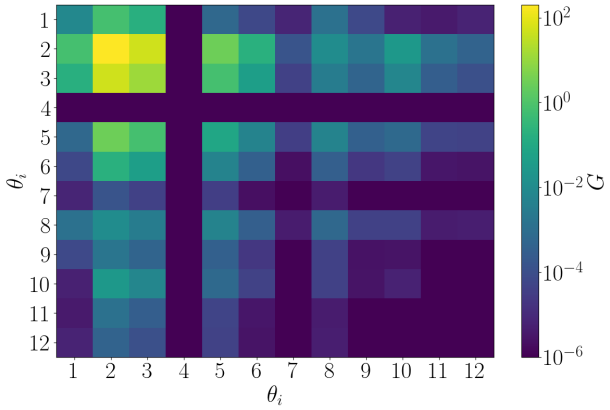

Figure 3: Empirical covariance of the loss gradients over 100 sampled points in input space. Figure presents the heatmap of $G = \|g\, g^T\|/100$, where $g \in \mathbb{R}^{100\times n}$ contains one gradient vector per row and n is the number of parameters to optimize. The blue–to–yellow colorbar encodes magnitude (blue ≈ 0, yellow $\gg 1$). The diagonal terms, G_{ii}, give each parameter's gradient variance, where the bright yellow cells flag parameters with the most substantial average influence on loss (parameters connected to the energy barrier considered). The off-diagonal terms, G_{ij} with $i \neq j$, show the absolute covariance between gradient components.

3.5.3 Hierarchical optimization

Building on the insights that our loss $\tilde{\Phi}(\theta)$ is ill-conditioned and that neither pure global nor pure local methods perform reliably solely, we adopt an adaptive strategy that leverages the problem's geometric structure. Informed by Sloppy Model Theory [23, 24, 25, 26], we reframe the search for a minimum as a sequence of steps guided by an iteratively refined geometric heuristic [27].

Geometric Model Injection via the *Gauss-Newton* Hessian

The heart of this heuristic is a local model of the objective loss function $\tilde{\Phi}$, that explicitly takes into account the fact that objective loss is dominated by the set of *stiff* directions, while the remaining *sloppy* modes contribute negligibly [23, 24, 25]. This decomposition is based on the information provided by the Hessian matrix of the objective loss. Under the assumption that the residuals, r_i (Eq.[37), are small, the objective loss' Hessian is well approximated by

$$[H(\theta_0)]_{ij} = [\nabla^2\tilde{\Phi}(\theta_0)]_{ij} \approx [H_{\mathrm{GN}}(\theta_0)]_{ij} = \sum_l \frac{\partial r_l}{\partial\theta_i}\frac{\partial r_l}{\partial\theta_j}, \tag{39}$$

where H_{GN} corresponds to the *Gauss-Newton* approximation. The full derivation of $H(\theta_0)$ is presented in Appendix [6.4]. The eigendecomposition of this matrix at an estimate θ_0,

$$H_{\mathrm{GN}}(\theta_0) = V\Lambda V^T, \quad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n) \tag{40}$$

reveals a near-universal property of these models [23, 26]. As shown in Fig.[4], the eigenvalue spectrum $\{\lambda_i\}$ decays exponentially, exposing a small set of *stiff* parameter combinations that dominate the model's behavior and a large set of *sloppy* combinations that have a negligible effect.

This justifies the hierarchical optimization approach. We partition the eigenpairs $\{(\lambda_i, v_i)\}$ of H_{GN} into two different subspaces. The *stiff* subspace $V_s = \mathrm{span}\{v_1, v_2, \ldots, v_k\}$ contains the large λ_i, while the *sloppy* subspace, V_l, is its orthogonal complement. Any parameter vector can be expressed as a deviation from the current estimate $\theta^{(t)}$ along these natural coordinates:

$$\theta = \theta^{(t)} + V_s\phi + V_l\psi. \tag{41}$$

where ϕ are the coordinates in the *stiff* subspace, while ψ are the coordinates in the *sloppy* subspace.

Heuristic Meta-Simulator ($\mathcal{H}_k$)

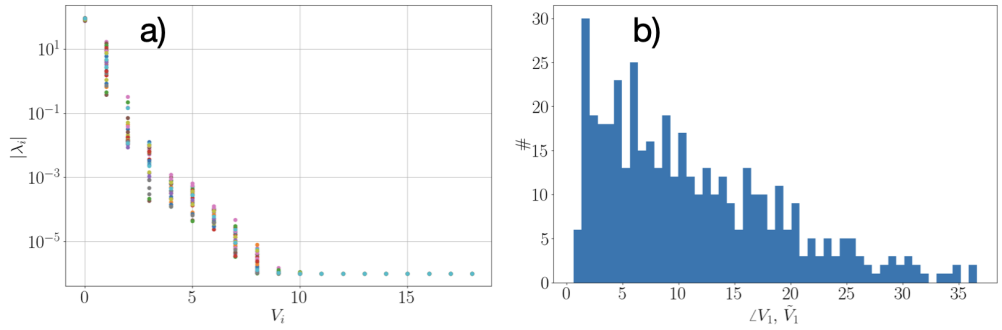

Figure 4: Figure presents the analysis of the *Gauss–Newton Hessian*, H_{GN} across input perturbations. (a) Spectrum of eigenvalues. For each of 30 points randomly sampled around the default condition (uniform isotropic noise), we form the $H_{\mathrm{GN}} = J^T J$, compute its eigenvalues $\{\lambda_i\}$, sort them in descending order, and plot$|\lambda_i|$ against index i (direction V_i). The rapid drop from λ_1 to λ_n highlights a few *stiff* parameter combinations versus many *sloppy* ones. (b) Angle-consistency histogram. For each sample, we take its leading eigenvector V_1. Then, we compute all pairwise angles between these V_1s (in degrees) and plot their histogram. A concentration of small angles shows that the dominant sensitivity direction is preserved across different input realizations.

This decomposition allows us to replace the original, intractable joint optimization with a more manageable, sequential one. We formalize this step as the application of the *heuristic meta*-simulator operator, $\mathcal{H}_t$. This operator maps the current parameter vector $\theta^{(t)}$ to the next, $\theta^{(t+1)}$, by executing a search constrained by the geometric model derived at $\theta^{(t)}$. The operation $\theta^{(t+1))} = \mathcal{H}_t(\theta^{(t)})$ is a composition of two optimization subproblems:

- *Stiff* step: $\theta^{(t')} = \theta^{(t)} + V_s^{(t)}\phi^*$, where $\phi^* = \arg\min_\phi \tilde{\Phi}(\theta^{(t)} + V_s^{(t)}\phi)$
- *Sloppy* step: $\theta^{(t+1)} = \theta^{(t')} + V_l^{(t)}\psi^*$, where $\psi^* = \arg\min_\psi \tilde{\Phi}(\theta^{(t')} + V_l^{(t)}\psi)$

This sequential procedure constitutes a single step guided by the local geometry of the manifold.

Iterative Refinement via Space Update

A static model based on the geometry at θ_k is insufficient to navigate the objective surface $\tilde{\Phi}$. The key to the method's power is the *iterative refinement* of the heuristic operator $\mathcal{H}_t$ in the style of *Manifold Boundary Approximation Method* [27]. After each step, the old operator $\mathcal{H}_t$ and its underlying basis V_t are discarded. A new heuristic $\mathcal{H}_{t+1}$ is then constructed by re-evaluating the geometry on the new point $\theta^{(t+1)}$. Hence, the overall optimization is a sequence of applications of these adaptive operators:

$$\theta^{(t+1)} = \mathcal{H}_t(\theta^{(t)}), \quad \theta^{(t+2)} = \mathcal{H}_{t+1}(\theta^{(t+1)}), \quad \ldots \tag{42}$$

This iterative process allows the algorithm to *see* the changing landscape and adjust the coordinate system, enabling it to follow the curved, *narrow valleys* characteristics of the *sloppy* model.

Convergence Analysis

The algorithm guarantees descent. Since each subproblem is solved by a descent method, the objective function is guaranteed to be non-increasing: $\tilde{\Phi}(\theta^{(t+1)}) \leq \tilde{\Phi}(\theta^{(t)})$. As the $\tilde{\Phi} > 0$, the sequence must converge.

As termination criteria, we consider the stability of the *stiff* subspace [27]. Convergence of the algorithm is assumed when the stiff subspace stops rotating, which is measured by analyzing the singular values $\{\sigma_i\}$ of the overlap matrix $V_s^{(t+1),T} V_s^{(t)}$. The misalignment, $\delta_s = \max_i(1 - \sigma_i)$, indicates convergence when it falls below a tolerance, $\delta_s < \epsilon_{\mathrm{space}}$.

The criterion ensures the converged point, θ_{final}, has (almost) the properties of a true minimum. If the algorithm has converged, it has found a stable region close to a local minimum θ_{min}. In this region, the gradient is well approximated by:

$$\nabla\tilde{\Phi}(\theta_{\mathrm{final}}) \approx H(\theta_{\mathrm{final}} - \theta_{\mathrm{min}}) = \sum_i \lambda_i v_i v_i^T (\theta_{\mathrm{final}} - \theta_{\mathrm{min}}) \tag{43}$$

The algorithm ensures that the final gradient lies in the *sloppy* subspace, which implies that the square norm of the gradient is bounded:

$$\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|_2 \leq \sqrt{N_{\text{sloppy}}} \cdot |\lambda_{\text{max sloppy}}| \cdot \|\theta_{\text{final}} - \theta_{\text{min}}\|_2. \tag{44}$$

Since $\lambda_{\text{max sloppy}}$, the largest eigenvalue (in magnitude) of the *sloppy* space is very small ($\ll 1$), the norm of the gradient at the converged point is guaranteed to be small. Appendix [6.6] presents the full derivation of Eq.[44].

The algorithm is a local method and does not guarantee finding the global minimum, θ_{global}. However, for *sloppy* models, the set of almost optimal parameters, $\mathcal{S}_\epsilon = \{\theta \in \mathbb{R}^N \mid \tilde{\Phi}(\theta) \leq \tilde{\Phi}(\theta_{\text{global}}) + \epsilon\}$ with $\epsilon > 0$, is a large, high-dimensional, extended region in the parameter space [23, 24, 25]. Its structure is analyzed by looking for the *sloppy* eigenvectors, $\{v_j\}$ of the Hessian at the minimum. As we consider deviations from the minimum, $\Delta\theta$, along the *sloppy* direction: $\Delta\theta = \sum_j \alpha_j v_j$, the change in the objective loss $\tilde{\Phi}$ is approximately given by:

$$\Delta\tilde{\Phi} \approx \frac{1}{2}\Delta\theta^T H \Delta\theta = \frac{1}{2}\sum_i \alpha_i v_i^T \sum_j \lambda_i v_j v_j^T \sum_k \alpha_k v_k. \tag{45}$$

Due to the orthonormality of the eigenvectors ($v_i^T v_j = \delta_{ij}$), this simplifies to:

$$\Delta\tilde{\Phi} \approx \sum_j \lambda_j \alpha_j^2. \tag{46}$$

Since all the sloppy eigenvalues λ_j are close to zero ($\ll 1$), we see that even large displacements in the parameter space produce negligible changes in the objective loss.

In this way, we notice that many different parameter vectors can produce a near-optimal fit, and while their components may differ from θ_{global}, the model's predictions are almost identical [23, 24, 25]. Furthermore, it mitigates the *curse of dimensionality* problem since the core optimization problem is reduced to a much smaller subspace.

4 Results

Now, we present the study of the scaling and performance of the approaches presented in Section 3.4.

We focus on the dataset of experimental conditions, which includes all experiments considered in [2] as well as data points from Tiago Dias' experimental measures [2]. In total, it corresponds to 225 data points where approximately 25% correspond to low pressure ($< 1\,\text{Torr}$) conditions. The dataset that concatenates all the sources can be obtained at the LINK.

We consider the surface kinetic scheme of O, O_2, and CO presented in [6]. The hyperparameters to optimize comprise a total of 29 parameters. It corresponds to four main groups:

- the parameters that appear in the desorption frequency $\nu_d = A + B * e^{E/(k_B T_w)}$ (following [2]) for O_F, $O_{2\,F}$ and CO_F species,
- steric factors of the reaction that involve the CO gas specie and CO_F and CO_S and do not involve metastables (CO Surface Kinetics Table 1 from [6]),
- steric factors of the reactions that dynamics of the metastables with CO without counting the metastable creation reactions (reactions 22 - 35 from Table 2 from [6]), the minimal energy for creating metastable states connected with $O_{2\,\text{fast}}$ and the minimal energy for destroying all the metastable states of the CO and O surface scheme.

We compare the different algorithms presented and can draw the following conclusions based on the results shown in Fig. [5]

The hierarchical optimization method (green line) demonstrates the fastest descent and the best final loss. Within the first approximately 50 iterations, it decreases from around $\tilde{\Phi} \approx 1.2$ to about 0.1. It then gradually improves to around 0.07 by the 400^{th} iteration. It indicates that stiff directional optimization quickly finds a favorable basin, while fine-tuning leads to the lowest loss. This method's success is likely due to its ability to navigate the complex, multi-modal, or sloppy nature of the optimization landscape, where a large number of parameters can be effectively grouped and optimized together before fine-tuning.

The CMA-ES algorithm (blue line) is robust to the initial guess but also the most expensive. It demonstrates a gradual and monotonic decrease in performance from approximately $\tilde{\Phi} \approx 1$ to around $\tilde{\Phi} \approx 0.3$. The differential evolution

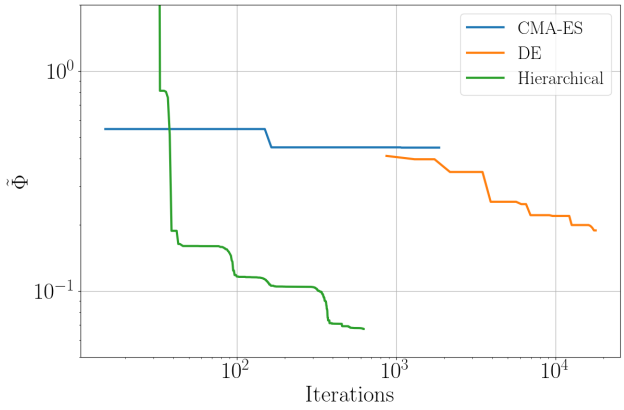

Figure 5: Optimization loss $\tilde{\Phi}$ vs. iteration count for the CMA-ES (blue), DE (orange) and Hierarchical optimization (red). The x-axis is the number of objective-function evaluations on a log scale. The y-axis is the total loss $\tilde{\Phi}$, also shown on a log scale to capture large dynamic range.

(DE) algorithm (orange line) continues this gradual decrease to a final loss of around $\tilde{\Phi} \approx 0.15$. These methods demonstrate themselves to be slow, and each iteration yields diminishing returns. This slow convergence is a hallmark of non-gradient-based methods when applied to complex, high-dimensional problems.

While the optimization provides the single best-fit parameters θ^* that minimize the loss function $\tilde{\Phi}$, due to the sloppy nature of the problem, there exists a *cloud* of alternative parameter sets that can describe the data *almost* equally well. By quantifying this uncertainty, we are also analyzing how well the data presented in the dataset constrain each model parameter that is optimized. The approach followed is based on the geometry of the loss function at the minimum [23, 27].

By expanding the loss function around the best fit up to the second order, we can define the confidence region, $\Delta\theta^T H(\theta^*)\Delta\theta \leq 2\Delta\tilde{\Phi}$, as the set of all parameters, $\theta = \theta^* + \Delta\theta$ for which the changes in the loss function are smaller than $\Delta\tilde{\Phi}$.

As a result, we estimate the uncertainty of each parameter, $|\Delta\theta_i|$ as the maximum value for the projection of the $\#\theta$-dimensional ellipsoid on that parameter's axis. It is given by:

$$|\Delta\theta_i| = \sqrt{2\,\Delta\tilde{\Phi}\,.\,\left(H_{GN}^{-1}\right)_{i\,i}} \tag{47}$$

The full derivation is presented in Appendix 6.6.

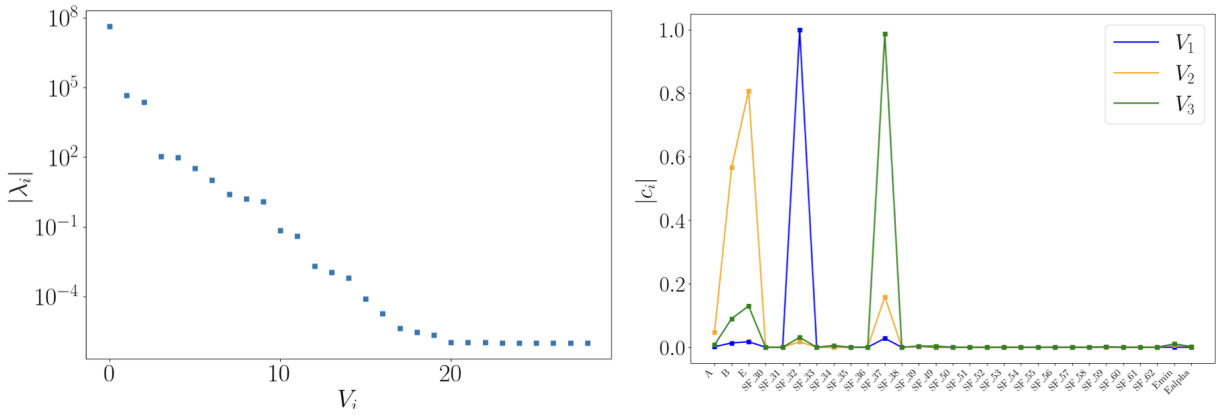

Figure 6: Eigen Features of the

The most important parameters for the optimization:

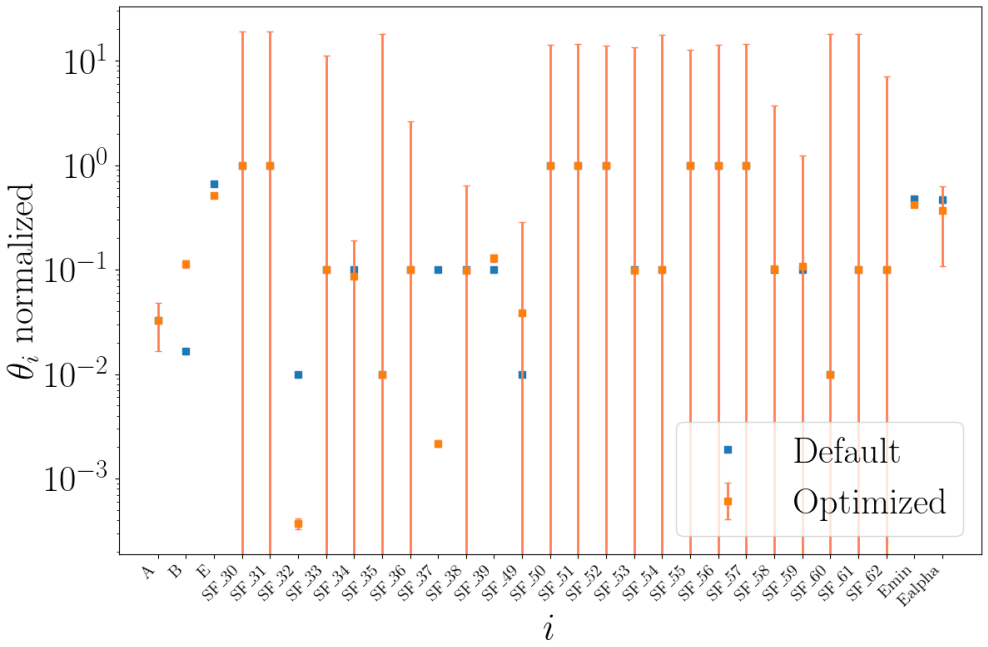

Figure 7: Scaled Paremeter Comparison, $\tilde{\Phi}_{\text{default}} = 0.1038$, $\tilde{\Phi}_{\text{opt}} = 0.0672$

- A, B, E: desorption reaction for O_F, $O_{2\,F}$ and CO_F
- $SF\_32 : CO + V_S \rightarrow CO_S$
- $SF\_37 : O + CO_S \rightarrow CO_2 + V_S$
- $SF\_39 : O_F + CO_S \rightarrow CO_2 + V_F + V_S$
- E_{min} and E_α for metastable states

5 Conclusions

We have addressed the challenge of determining surface recombination kinetics in the presence of experimental uncertainty. Our approach was twofold: first, we framed the problem within a probabilistic framework to derive a principled objective loss function; second, we introduced a hierarchical optimization algorithm designed to navigate the sloppy parameter space inherent to these models efficiently. The comparative analysis confirms that our algorithm is not only more computationally efficient but also yields more accurate and reliable kinetic parameters than standard global optimization methods. In future work, we plan to expand this analysis to additional plasma-surface systems and various other operating regimes.

References

[1] Vasco Guerra. Analytical model of heterogeneous atomic recombination on silicalike surfaces. *IEEE Transactions on Plasma Science*, 2007.

[2] Pedro Viegas et al. Surface recombination in pyrex in oxygen dc glow discharges: mesoscopic modelling and comparison with experiments. *Plasma Sources Science and Technology*, 2024.

[3] Vasco Guerra José Afonso, Luca Vialetto and Pedro Viegas. Plasma-induced reversible surface modification and its impact on oxygen heterogeneous recombination. *Journal of Physics D: Applied Physics*, 2024.

[4] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 1997.

[5] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

[6] Blandine Berdugo. Surface recombination on pyrex in co2 glow discharges. 2024.

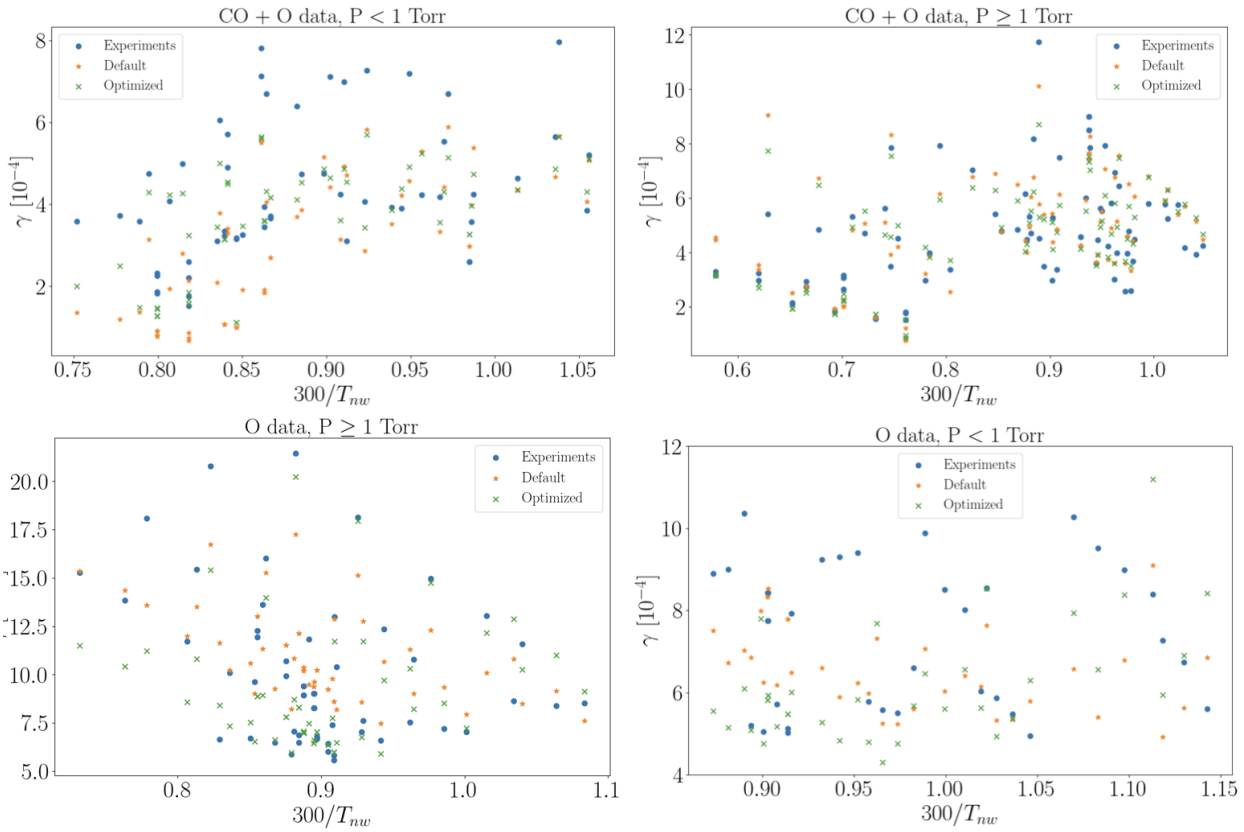

Figure 8: Results

[7] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[8] David J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[9] Max Welling Diederik P. Kingma. *An Introduction to Variational Autoencoders*. Foundations and Trends in Machine Learning, 2019.

[10] A Tejero del-Caz et al. The lisbon kinetics boltzmann solver. *Plasma Sources Science and Technology*, 2019.

[11] Donald R. Jones et al. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization 13: 455–492*, 1998.

[12] Eric Brochu et al. Geodesic acceleration and the small-curvature approximation for nonlinear least squares, 2024.

[13] Swagatam Das and Ponnuthurai Nagaratnam Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, 2011.

[14] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[15] Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

[16] John A Nelder and Roger Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[17] Michael JD Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[18] Jorge Nocedal. Updating quasi newton matrices with limited storage. *Mathematics of Computation*, 35(151):951–958, July 1980.

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep

learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc., 2019.

[20] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[21] Zsolt et al. Ugray. Scatter search and local nlp solvers: A multistart framework for global optimization. *INFORMS Journal on Computing*, 2007.

[22] Rajesh Shrestha. Natural gradient methods: Perspectives, efficient-scalable approximations, and analysis, 2023.

[23] Mark K. Transtrum, Benjamin B. Machta, and James P. Sethna. Why are nonlinear fits to data so challenging? *Physical Review Letters*, 104(6), February 2010.

[24] Mark K. Transtrum, Benjamin B. Machta, and James P. Sethna. Geometry of nonlinear least squares with applications to sloppy models and optimization. *Physical Review E*, 83(3), March 2011.

[25] Mark K. Transtrum and James P. Sethna. Geodesic acceleration and the small-curvature approximation for nonlinear least squares, 2012.

[26] Mark K. Transtrum, Benjamin Machta, Kevin Brown, Bryan C. Daniels, Christopher R. Myers, and James P. Sethna. Sloppiness and emergent theories in physics, biology, and beyond, 2015.

[27] Mark K Transtrum, Bryan B Machta, and James P Sethna. An efficient, geometrically motivated algorithm for fitting sloppy models. *Physical Review E*, 83(3):036701, 2011.

[28] David Freeborn. Sloppy models, renormalization group realism, and the success of science. *Erkenntnis*, 90:645–673, 08 2023.

[29] Michael J. Evans and Jeffrey S. Rosenthal. *Probability and Statistics: The Science of Uncertainty*. W. H. Freeman and Company, New York, 2nd edition, 2010.

6 Appendices

6.1 Observed-data likelihood derivation

Here we present the detailed derivation presented in Section 3.2. We start from the general *observed-data* distribution:

$$p\big(\gamma_i^{\exp}, x_i^{\exp} \mid \theta\big) \;=\; \int p(\gamma_i^{\exp} \mid x_i^*, \theta)\, p(x_i^{\exp} \mid x_i^*)\, dx_i^*, \tag{48}$$

As explained in the main text, since $\gamma(x, \theta)$ is nonlinear, the general expression for $p\big(\gamma_i^{\exp}, x_i^{\exp} \mid \theta\big)$ has no closed form. Hence, we will use the *Gaussian* approximation.

We start by assuming that $\gamma(x^*, \theta)$ is smooth in x^*, we do its first-order Taylor expansion around x^{exp}:

$$\gamma(x^*, \theta) \approx \gamma(x^{\exp}, \theta) + J_x(\theta)\delta x, \quad J_x(\theta) = \nabla_x \gamma(x, \theta)\mid_{x^{\exp}} \tag{49}$$

where $\delta x = x^* - x^{\exp}$.

Property: An affine transformation applied to a random Gaussian variable is still Gaussian [29].

In the case of δx, we have:

$$\mathbb{E}[\delta x] = \mathbb{E}[x^*] - \mathbb{E}[x^{\exp}] = 0 \tag{50}$$

$$\begin{aligned}\Sigma_{ij}^{\delta x} &= \mathbb{E}[\delta x_i\, \delta x_j] \\ &= \mathbb{E}[(x^* - x^{\exp})_i\, (x^* - x^{\exp})_j] \\ &= x_i^* x_j^* + \mathbb{E}[x_i^{\exp}\, x_j^{\exp}] - 2\, x_i^*\, \mathbb{E}[x_j^{\exp}] \\ &= \mathbb{E}[x_i^{\exp} x_j^{\exp}] - \mathbb{E}[x_i^{\exp}]\, \mathbb{E}[x_j^{\exp}] = \Sigma_{ij}^{x^{\exp}}\end{aligned} \tag{51}$$

which allows to conclude that:

$$\delta x \sim \mathcal{N}(0, \Sigma^{x^{\exp}}) \tag{52}$$

For the case of $J\delta x$, we have:

$$\mathbb{E}[J_i^m \delta x_m] = J_i^m \mathbb{E}[\delta x_m] = 0 \tag{53}$$

$$\begin{aligned}\Sigma_{ij}^{J\delta x} &= \mathbb{E}[(J_i^m \delta x_m)(J_j^n \delta x_n)] = J_i^m J_j^n \mathbb{E}[\delta x_m \delta x_n] \\ &= J_i^m J_j^n \Sigma_{mn}^{\delta x} = J_i^m \Sigma_{mn}^{\delta x} J_j^{T\, n} = (J \Sigma^{x^{\exp}} J^T)_{ij}\end{aligned} \tag{54}$$

which implies:

$$J\delta x \sim \mathcal{N}(0, J_x \Sigma_x J_x^T) \tag{55}$$

Property: A finite sum of independent Gaussian random variables, $S = \sum_i X_i, \;\; X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, is Gaussian and given by $S \sim \mathcal{N}(\sum_i \mu_i, \sum_i \sigma_i^2)$ [29].

Since γ^{exp} is given by:

$$\gamma^{\text{exp}} = \gamma(x^*, \theta) + \varepsilon \approx \gamma(x^{\text{exp}}, \theta) + J_x(\theta)\delta x_i + \sigma_y \varepsilon. \tag{56}$$

where δx_i and ε are two independent Gaussian random variables, we can conclude that:

$$\gamma_i^{\text{exp}} \mid x_i^{\text{exp}}, \theta \sim \tilde{p}(\gamma_i^{\text{exp}} \mid x_i^{\text{exp}}, \theta) = \mathcal{N}(\mu_i, V_i) \tag{57}$$

with

$$\mu_i = \gamma(x_i^{\text{exp}}, \theta) \;\; \text{and} \;\; V_i = \sigma_{y,i}^2 + J_{x,i}(\theta)\Sigma_{x,i}J_{x,i}^T(\theta) \tag{58}$$

Leveraging on the previous calculations and on the *Gaussian* approximation, we can approximate Eq. [48] as:

$$p(\gamma_i^{\text{exp}}, x_i^{\text{exp}} \mid \theta) \approx \tilde{p}(\gamma_i^{\text{exp}} \mid x_i^{\text{exp}}, \theta) \int dx_i^* \, p(x_i^{\text{exp}} \mid x_i^*) \tag{59}$$

and the negative *log-likelihood* corresponds to:

$$\mathcal{L}(\theta) = \sum_{i=1}^{M} \log p(x_i^{\text{exp}}, \gamma_i^{\text{exp}} | \theta) \approx \sum_{i=1}^{M} \log \tilde{p}(\gamma_i^{\text{exp}} \mid x_i^{\text{exp}}, \theta) + \text{const}_1 \tag{60}$$

where

$$\text{const}_1 = \log \int dx_i^* \, p(x_i^{\text{exp}} \mid x_i^*), \tag{61}$$

and it is independent w.r.t θ.

As a result, we recover $\mathcal{L}_{\text{approx}}(\theta)$:

$$\mathcal{L}_{\text{approx}}(\theta) = -\frac{1}{2} \sum_{i=1}^{M} \left[\frac{(\gamma_i^{\text{exp}} - \mu_i)^2}{V_i} + \log V_i \right] \tag{62}$$

where we neglect all the terms independent w.r.t θ.

6.2 $p(\delta x \mid \gamma^{\text{exp}}, \theta)$ derivation

We start from Eq.[30], which we repeat here for convenience:

$$p(x^* | x^{\text{exp}}, \gamma^{\text{exp}}, \theta) \propto \exp -\frac{1}{2}\delta x^T \Sigma_x^{-1} \delta x - \frac{1}{2\sigma_y^2}(\gamma^{\text{exp}} - \mu - J_x \delta x)^2 \tag{63}$$

Now we expand the second term, which allows us to obtain:

$$\begin{aligned} p(x_i^* | x_i^{\text{exp}}, \gamma_i^{\text{exp}}, \theta) &\propto \exp -\frac{1}{2}\delta x^T \Sigma_x^{-1} \delta x - \frac{1}{2\sigma_y^2}\delta x^T J_x^T J_x \delta x - \frac{1}{2\sigma_y^2}\delta x^T J_x^T (\gamma^{\text{exp}} - \mu) - \frac{1}{2\sigma_y^2}(\gamma^{\text{exp}} - \mu)^2 \\ &\propto \exp -\frac{1}{2}\delta x^T \left(\Sigma_x^{-1} + \frac{1}{\sigma_y^2} J_x^T J_x \right) \delta x - \delta x^T \frac{1}{\sigma_y^2} J_x^T (\gamma^{\text{exp}} - \mu) \end{aligned} \tag{64}$$

Using the general expression for the square completion, we obtain the expression presented in Eqs. [31 - 32].

6.3 Analitic Gradient derivation $\nabla \tilde{\Phi}$

Considering the $\tilde{\Phi}(\theta)$ and $\gamma_m(\theta)$ where m corresponds to an input experimental condition, we define:

$$\gamma_m(\theta) = \sum_k [T_1(\theta)]_k y^*(\theta)_k + \sum_{kl} [T_2(\theta)]_{kl} y^*(\theta)_k y^*(\theta)_l \tag{65}$$

which corresponds to the general functional expression depicted on Eq.[12]. Moreover, y^* corresponds to the steady-state vector of the chemical densities (as presented in Section 2.3) and it corresponds to *fixed point* of the physical model's system of equations

$$\vec{F}(\vec{x}, \vec{y^*}(\vec{x}, \theta); \theta) = \vec{0}, \tag{66}$$

as presented in Eq.[9]. The $T_1(\theta)$ and $T_2(\theta)$ correspond to tensors whose entries include the reaction rates that participate directly for the recombination probability.

Now, we compute the analytical expression for $\nabla\tilde{\Phi}(\theta)$, given by the following expression in Euclidean space:

$$\nabla\tilde{\Phi}(\theta) = \sum_i \frac{\partial\tilde{\Phi}(\theta)}{\partial\theta_i}\hat{\mathbf{e}}_i \tag{67}$$

with

$$\frac{\partial\tilde{\Phi}(\theta)}{\partial\theta_i} = \sum_{m=1}^{M} \frac{1}{(\gamma_m^{\exp})^2}(\gamma_m(\theta) - \gamma_m^{\exp})\frac{\partial\gamma_m(\theta)}{\partial\theta_i} \tag{68}$$

which can be further simplified if we use Eq.[65]. Thus, we get

$$\begin{aligned}\frac{\partial\gamma_m(\theta)}{\partial\theta_i} = &\sum_k \left(\frac{\partial}{\partial\theta_i}[T_1(\theta)]_k\right) y_k^* + \sum_{ln}\left(\frac{\partial}{\partial\theta_i}[T_2(\theta)]_{ln}\right) y_l^* y_n^* \\ &+ \sum_k [T_1(\theta)]_k \frac{\partial y_k^*}{\partial\theta_i} + \sum_{ln}\left[T_2(\theta) + T_2^T(\theta)\right]_{ln} y_l^* \frac{\partial y_n^*}{\partial\theta_i}\end{aligned} \tag{69}$$

Using the *Implicit Differentiation* Theorem:

$$\frac{dF_i}{d\theta_m} = 0 \Leftrightarrow \frac{\partial F_i}{\partial\theta_m} + \sum_n \frac{\partial F_i}{\partial y_n}\frac{\partial y_n}{\partial\theta_m} = 0 \tag{70}$$

$$\Leftrightarrow [\partial_\theta F]_{im} + \sum_n [\partial_y F]_{in}[\partial_\theta y]_{nm} = 0 \tag{71}$$

$$\Rightarrow [\partial_\theta F] + [\partial_y F].[\partial_\theta y] = \mathbf{0} \tag{72}$$

where we have that: $[\partial_\theta F] \in \mathbb{R}^{d_F \times d_\theta}$, $[\partial_y F] \in \mathbb{R}^{d_F \times d_y}$ and $[\partial_\theta y] \in \mathbb{R}^{d_y \times d_\theta}$.

While $d_F = d_y$, in general $d_y \neq d_\theta$. The more interesting cases correspond to $d_\theta > d_y$, where we have a large set of parameters to optimize. However, it also implies that there are more parameters (d_θ) than equations (d_F). Thus, each column-wise system $\sum_j [\partial_y F]_{nj}[\partial_\theta y]_{j,:} = -[\partial_\theta F]_{n,:}$ is underdetermined. Geometrically, this means that $[\partial_\theta F]$ has a nontrivial null space of dimension at least $d_\theta - d_F$ and any component of $[\partial_\theta y]$ lying in that null space does not change F at first order.

Hence, in the most interesting cases, $[\partial_\theta F]$ is not full rank and we obtain the solution for $[\partial_\theta y]$ by solving the **least-squares** problem:

$$\min_Y \|[\partial_y F] \cdot Y + [\partial_\theta F]\|_F^2 \tag{73}$$

where $Y = [\partial_\theta y^*]$ and $\|\cdot\|_F$ is the Frobenius norm. By selecting the solution with the smallest Frobenius norm, we choose the solution that introduces minimal sensitivity in the directions of the steady-state equations' constraints and we fix the remaining at zero [REF].

As a result, by combining the results of Eq.[67,68, 69 and 72], we can compute $\nabla\tilde{\Phi}(\theta)$ in a closed and efficient way, where we use the AD framework to compute $[\partial_y F]$ and $[\partial_\theta F]$ analytically.

6.4 Derivation $\nabla^2\tilde{\Phi}(\theta)$

Here, we present the full derivation *Hessian* matrix of the objective loss, $\nabla^2\tilde{\Phi}$. As computed in Section 6.3, $\partial\tilde{\Phi}/\partial\theta_i$ is given by:

$$\frac{\partial\tilde{\Phi}}{\partial\theta_i} = \sum_{l=1}^{M} r_l(\theta)\frac{\partial r_l(\theta)}{\partial\theta_i} = \sum_{l=1}^{M} r_l(\theta) J_{li}(\theta) \tag{74}$$

with $J_{li}(\theta) = \partial r_l/\partial\theta_i$. Thus, $\nabla\tilde{\Phi}$ can be written as

$$\nabla\tilde{\Phi}(\theta) = J^T(\theta)\mathbf{r}(\theta) \tag{75}$$

Regarding $[H(\theta)]_{ij}$, we have:

$$[H(\theta)]_{ij} = \frac{\partial}{\partial\theta_i}\frac{\partial\tilde{\Phi}}{\partial\theta_j} = \sum_l \frac{\partial r_l}{\partial\theta_i}\frac{\partial r_l}{\partial\theta_j} + \sum_l r_l \frac{\partial^2 r_l}{\partial\theta_i\partial\theta_j} \tag{76}$$

and so $H(\theta)$ is given by:

$$H(\theta) = \nabla^2 \tilde{\Phi}(\theta) = J^T J + S, \quad S = \sum_l r_l \nabla^2 r_l \tag{77}$$

Now, using the diagonalization of $H_{\mathrm{GN}} = J^T J$ presented in Eq. [40] and for each *stiff* mode v_i (large λ_i), we have:

$$v_i^T H(\theta) v_i = v_i^T (J^T J + S) v_i = \lambda_i + v_i^T S v_i. \tag{78}$$

We know that as long as the residuals r_i are small (our initial guess θ_0 is good enough), we have:

$$|v_i^T S v_i| \leq \|S\|_2 = \left\| \sum_l r_l \nabla^2 r_l \right\|_2 \ll \lambda_i. \tag{79}$$

Hence, the approximation $H(\theta) \approx H_{\mathrm{GN}}(\theta)$ is well justified.

6.5 Hierarchical Optimization Algorithm Derivation

We start from the second order Taylor expansion of the objective loss around an initial guess θ_0:

$$\tilde{\Phi}(\theta_0 + \Delta\theta) = \tilde{\Phi}(\theta_0) + \nabla \tilde{\Phi}^T(\theta_0) \Delta\theta + \frac{1}{2} \Delta\theta^T \nabla^2 \tilde{\Phi}(\theta_0) \Delta\theta \tag{80}$$

Under the approximation that the residuals are small (θ_0 is a good estimate), we can use the approximations presented in Eq. [75,77] and rewrite $\tilde{\Phi}(\theta_0 + \Delta\theta)$ as

$$\tilde{\Phi}(\theta_0 + \Delta\theta) \approx \tilde{\Phi}(\theta_0) + \frac{1}{2} \Delta\theta^T H_{\mathrm{GN}}(\theta_0) \Delta\theta \tag{81}$$

Taking advantage of H_{GN} being positive semidefinite, we obtain its eigendecomposition, as firstly presented in Eq.[40], and we get:

$$H_{\mathrm{GN}} = V \Lambda V^T, \quad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad \lambda_1 \geq \lambda_2 \geq \cdots \geq 0 \tag{82}$$

By choosing an integer k, we do the space partition given by:

$$V = [V_s \;\; V_l], \quad \Lambda = \begin{bmatrix} \Lambda_s & 0 \\ 0 & \Lambda_l \end{bmatrix} \tag{83}$$

where

- $V_s \in \mathbb{R}^{n \times k}$, $\Lambda_s = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$ - *stiff* modes,
- $V_l \in \mathbb{R}^{n \times (n-k)}$, $\Lambda_l = \mathrm{diag}(\lambda_{k+1}, \ldots, \lambda_n)$ - *sloppy* modes,

The choice of k is based on the usual PCA choice and is given by:

$$\sum_{i=1}^{k} \lambda_i > \sigma_{\min} \sum_{i=1}^{n} \lambda_i \tag{84}$$

and $\sigma_{\min}$ corresponds to a hyperparameter and is assumed $\sigma_{\min} = 0.9$.

In the case that a set of $\tilde{k}$ eigenvalues of Λ_l, $\{\lambda_l\}$ verifies $\lambda_l < \varepsilon$, where ε corresponds to a minimal threshold, V_l is replaced by the *reduced sloppy* subspace, $\tilde{V}_l \in \mathbb{R}^{n \times (n-k-\tilde{k})}$, $\Lambda_l = \mathrm{diag}(\lambda_{k+1}, \ldots, \lambda_{n-\tilde{k}})$.

Leveraging on the new space decomposition, we rewrite the objective problem presented in Eq.[36] as

$$\min_{\phi, \psi} \tilde{\Phi}(\theta_0 + V_s \phi + V_l \psi) \tag{85}$$

where ψ are the new coordinates connected to the *stiff* subspace and ψ are the coordinates from the *sloppy* subspace.

However, this formulation presents two main flaws. On the one hand, while the V_s and V_l decompositions are always applicable, they are more valuable when the current best estimate for the parameters $\theta^{(t)}$ is *close* from θ_0.

On the other hand, as we perform $\min_{\phi,\psi}$ in a joint way, we still suffer from the ill-conditioning and possible *curse of dimensionality*. Having in mind these flaws and relaxing a bit the problem, we present Algorithm 1.

TABLE DEFAULT HYPERPARAMETERS

Algorithm 1 Hierarchical Optimization

Require: Objective loss $\tilde{\Phi}(\cdot)$ function, initial guess $\theta^{(0)}$, max iters $T_{\max}$, stopping criteria $\varepsilon_{\text{stop}}$, $\text{criteria}(\cdot)$ and $\text{eigendecompose}(\cdot)$ functions

// Initialize
Compute $H_{\text{GN}}(\theta^{(0)})$
$V_s^{(0)}, V_l^{(0)} \leftarrow \text{eigendecompose}\left(H_{\text{GN}}(\theta^{(0)})\right)$
$t \leftarrow 1$
while $t < T_{\max}$ **do**
    // 1) Stiff-subspace solve

$$\phi^{(t)} \leftarrow \arg\min_{\phi} \tilde{\Phi}\left(\theta^{(t-1)} + V_s^{(t-1)}\phi + V_l^{(t-1)}\psi^{(t-1)}\right)$$

    // 2) Sloppy-subspace solve

$$\psi^{(t)} \leftarrow \arg\min_{\psi} \tilde{\Phi}\left(\theta^{(t-1)} + V_s^{(t-1)}\phi^{(t)} + V_l^{(t-1)}\psi\right)$$

    // 3) Update full parameter

$$\theta^{(t)} \;=\; \theta^{(t-1)} + V_s^{(t-1)}\,\phi^{(t)} + V_\ell^{(t-1)}\,\psi^{(t)}$$

    // 4) Update space decomposition
    Compute $H_{\text{GN}}(\theta^{(t)})$
    $V_s^{(t)}, V_l^{(t)} \leftarrow \text{eigendecompose}\left(H_{\text{GN}}(\theta^{(t)})\right)$
    // 5) Check stopping
    if $\text{criteria}\left(\theta^{(t-1)}, \theta^{(t)}, V_s^{(t-1)}, V_s^{(t)}\right) < \varepsilon_{\text{stop}}$ **then**
        break
    end if
    $t \leftarrow t + 1$
end while
return $\theta^{(t)}$

6.6 Derivation of the Upper Bound for $\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|_2$

Using the algorithm and having successfully achieved the convergence confirmation, we have that the subspace stabilization criterion ($V_s^{(T-1)} \approx V_s^{(T)}$) is fulfilled and that the *stiff* optimization was enforced, which allows us to conclude that $P_s^{(T)}\nabla\tilde{\Phi}(\theta_{\text{final}}) \approx 0$, with $P_s = V_s\,V_s^T$ and $P_l = V_l\,V_l^T$.

Moreover, since the projection operators form a complete basis ($P_s + P_l = I$), the final gradient must lie entirely in the sloppy subspace: $\nabla\tilde{\Phi}(\theta_{\text{final}}) = P_s\nabla\tilde{\Phi}(\theta_{\text{final}})$

Now, we analyze the norm of the sloppy gradient component using the quadratic approximation around the local minimum, $\theta_{\min}$, that the algorithm has found and we have that:

$$\nabla\tilde{\Phi}(\theta_{\text{final}}) \approx H(\theta_{\min})(\theta_{\text{final}} - \theta_{\min}). \tag{86}$$

Projection onto the sloppy subspace and expressing the Hessian in its eigenbasis gives:

$$\nabla\tilde{\Phi}(\theta_{\text{final}}) = \sum_{j\in\text{sloppy}} \lambda_j v_j v_j^T(\theta_{\text{final}} - \theta_{\min}) = \sum_{j\in\text{sloppy}} c_j v_j \tag{87}$$

where we defined the scalar components $c_j = \lambda_j v_j^T(\theta_{\text{final}} - \theta_{\min})$.

Taking advantage of the orthonormality of the eigenvectors $\{v_j\}$, we have:

$$\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|_2^2 = \sum_{j\in\text{sloppy}} c_j^2. \tag{88}$$

Now, we can bound each component c_j^2 using the *Cauchy-Schwartz* inequality:

$$|c_j| = |\lambda_j| \cdot |v^T(\theta_{\text{final}} - \theta_{\min})| \leq |\lambda_j| \cdot \|v_j\| \cdot \|\theta_{\text{final}} - \theta_{\min}\| \tag{89}$$

Since $\|v_j\|_2 = 1$, c_j^2 is bounded by:

$$c_j^2 \leq \lambda_j^2 \cdot \|\theta_{\text{final}} - \theta_{\min}\|_2^2 \tag{90}$$

Replacing this into the sum and letting $\lambda_{\max\ \text{sloppy}}$ be the largest (in magnitude) of the sloppy eigenvalues, we get the final bound:

$$\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|_2^2 \leq \sum_{j\in\text{sloppy}} \lambda_{\max\ \text{sloppy}}^2 \cdot \|\theta_{\text{final}} - \theta_{\min}\|_2^2 = N_{\text{sloppy}} \cdot \lambda_{\max\ \text{sloppy}}^2 \cdot \|\theta_{\text{final}} - \theta_{\min}\|_2^2. \tag{91}$$

Finally, by taking the square root, we get the following upper bound for $\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|$:

$$\|\nabla\tilde{\Phi}(\theta_{\text{final}})\|_2 = \sqrt{N_{\text{sloppy}}} \cdot |\lambda_{\max\ \text{sloppy}}| \cdot \|\theta_{\text{final}} - \theta_{\min}\|_2 \tag{92}$$

6.7 Effective model

Starting from Eq.[81]with $\theta_0 = \theta_{\min}$ and by using its *stiff* and *sloppy* decomposition, we have:

$$\tilde{\Phi}(\theta_0 + \Delta\theta) \approx \tilde{\Phi}(\theta_0) + \frac{1}{2}\phi^T\Lambda_{\text{s}}\phi + \frac{1}{2}\psi^T\Lambda_{\text{l}}\psi \tag{93}$$

where we used $\Delta\theta = V_s\phi + V_l\psi$.

$$\tilde{\Phi}_{\text{eff}}(\phi) = \min_{\psi}\tilde{\Phi}(\phi_0 + V_s\phi + V_l\psi) \approx \tilde{\Phi}(\phi_0 + V_s\phi) = \tilde{\Phi}(\theta_0) + \frac{1}{2}\phi^T\Lambda_{\text{s}}\,\phi \tag{94}$$

As a result, we see that the *effective* number of parameters, p_{eff} lives in $\mathbb{R}^k$, with $k \ll n$, as the *sloppy* modes have a negligible effect.

$$p_{\text{eff}} = k = \#\{\lambda_i : \lambda_i > \text{cutoff}\} \tag{95}$$

Representing the vector of hyperparameters to optimize $|\theta\rangle$ as

$$|\theta\rangle = \sum_{i=0}^{n}|\theta_i\rangle = \sum_{i=0}^{l}|E_i\rangle + \sum_{i=l+1}^{n}|SF_i\rangle \tag{96}$$

and by noticing that the *stiff* and *sloppy* modes $|v_\alpha\rangle$ and their corresponding eigenvalues λ_α are derived from

$$H_{\text{GN}}(\theta_0)\,|v_\alpha\rangle = \lambda_\alpha\,|v_\alpha\rangle\,, \tag{97}$$

we can express $|v_\alpha\rangle$ in θ basis as

$$|v_\alpha\rangle = \sum_{\theta}|\theta\rangle\,\langle\theta|v_\alpha\rangle = \sum_{\theta} c_{\theta,\alpha}\,|\theta\rangle\,. \tag{98}$$

Hence, $|v_\alpha\rangle$ modes are linear combinations of energy components and sterical factors hyperparameters. *Stiff* modes are combinations that are highly correlated with the measured recombination probabilities, while the *sloppy* ones are related to possible redundancies in the physical system (the energies are defined up to a constant; only ratios of sterical factors are important, . . .)

6.8 Uncertainty estmation $|\Delta\theta_i|$

We want to estimate $|\Delta\theta_i|$ around the optimized parameter conditions θ^*.

$$\mathcal{L}(\Delta\theta_i, \mu) = (\Delta\theta_i)^2 + \mu\,.\left(\Delta\theta_m H_{GN}^{mj}(\theta^*)\Delta\theta_j - 2\Delta\tilde{\Phi}\right) \tag{99}$$

$$\begin{aligned}\nabla_{\Delta\theta_i}\mathcal{L} = 0 \implies 2\,\Delta\theta_i &= 2\,\mu H_{GN}^{ij}\Delta\theta_j \\ \Leftrightarrow \Delta\theta_j &= \frac{\Delta\theta_i}{\mu}(H_{GN}^{-1})_{ji}\end{aligned} \tag{100}$$

From the constraint we get:

$$\left(\frac{\Delta\theta_i}{\mu}\right)^2 (H_{GN}^{-1}e_i)^T H_{GN}\,(H_{GN}^{-1}e_i) = 2\Delta\tilde{\Phi}$$

$$\Leftrightarrow \left(\frac{\Delta\theta_i}{\mu}\right)^2 e_i^T H_{GN}^{-1} e_i = 2\Delta\tilde{\Phi} \Rightarrow \frac{1}{\mu} = \frac{1}{\Delta\theta_i}\frac{\sqrt{2\,\Delta\tilde{\Phi}}}{(H_{GN}^{-1})_{ii}} \tag{101}$$

Which allows one to conclude that:

$$\Delta\theta_i = \sqrt{2\,\Delta\tilde{\Phi}\,.\,(H_{GN}^{-1})_{ii}} \tag{102}$$