Inverse learning of physical constitutive relations: PDE constrained optimization

Dean Deng, Jueyi Liu, Rui Yan

March 2020

1 Introduction

Continuum level coarse-grained models rely on the parameterization of constitutive equations, relations that mathematically map two physical quantities such as stress-strain and composition-strain to an intrinsic materials property [2]. Using such constitutive relation, a forward model allows for the prediction of the desired physical properties by solving the relevant partial differential equations (PDEs) such as the continuity equation at the length scale of interest. For example, in a typical Li-ion battery material system, knowing the lithium composition-strain and stress-strain relation allows one to predict the elastic strain field distribution inside battery particles, which is relevant to battery life time studies [5, 6].

Existing methods for obtaining the "correct" parameters for these constitutive relations face severe challenges: bottom-up first-principle calculations are accurate but can be computationally expensive, while direct experimental measurement are often hindered by technical limitation. Recent research advancement in X-ray, optical and electron microscopy has provided scientists with a wealth of image data. Inside the image, the pixels are spatially correlated via PDEs that inherently embed constitutive relations of interest, thus offering an alternative to the aforementioned challenge [7].

To address the challenges mentioned above, in this work, we provide a general inverse learning framework to extract inherent physical constitutive relations by solving a backward PDE constrained optimization problem. Specifically, the approach is demonstrated in a battery material, where both scientific and industrial interests are high. First the PDE constraint was transformed into a system of algebraic equations via finite element discretization [9]. Two constitutive relations were then explored: (1) lithium composition-strain relation, parameterized by coefficients in the Legendre polynomial series, and (2) stress-strain relation, parameterized by the stiffness tensor. We show that the first relation learning can be transformed into a unconstrained linear least squares problem with a unique solution while the second one is non-convex, with multiple optimal solutions. Due to the non-linear non-convex nature of the second problem, steepest descent, conjugate gradient and Newton's method were investigated, and an empirical suggestion on reaching local minimum was provided.

2 Problem Statement

We briefly review the forward model, then state the inverse problem and frame it as a PDE constrained optimization problem.

2.1 Math Preliminaries

We first introduce variables: x_m, $\boldsymbol{d}$, and $\boldsymbol{e}$. Here, for $\boldsymbol{r} = (x; y) \in \mathbb{R}^2$, $x_m(\boldsymbol{r}) \in [0, 1]$ represents the lithium spatial distribution; $\boldsymbol{d}(\boldsymbol{r}) := (u(\boldsymbol{r}); v(\boldsymbol{r})) \in \mathbb{R}^2$ represents the displacement vector for each position; $\boldsymbol{e}$ represents the strain tensor, capturing the spatial derivative of $\boldsymbol{d}$, with the following relation:

$$\boldsymbol{e} = \begin{bmatrix} e_{xx} & e_{xy} \\ e_{yx} & e_{yy} \end{bmatrix} = \frac{1}{2}(\nabla \boldsymbol{d} + \nabla \boldsymbol{d}^T) = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{1}{2}\frac{\partial u}{\partial y} + \frac{1}{2}\frac{\partial v}{\partial x} \\ \frac{1}{2}\frac{\partial u}{\partial y} + \frac{1}{2}\frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{bmatrix} \tag{1}$$

By definition, the strain tensor is symmetric and can be decomposed into 2 parts: elastic strain and chemical strain:

$$e = e^{chem} + e^{el} \tag{2}$$

Each component follows a constitutive relation. The chemical strain is composition dependent, with a strain-composition relation denoted as f:

$$e^{chem} = \begin{bmatrix} e^{chem}_{xx} \\ e^{chem}_{xy} \\ e^{chem}_{yy} \end{bmatrix} = \begin{bmatrix} f_x(x_m) \\ 0 \\ f_y(x_m) \end{bmatrix} = f(x_m) \tag{3}$$

The elastic strain-stress relation is linear and can be parameterized by a stiffness tensor $\boldsymbol{C}(\alpha, \beta, \gamma, \omega)$ as follows,

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{yy} \end{bmatrix} = \boldsymbol{C} \begin{bmatrix} e^{el}_{xx} \\ e^{el}_{xy} \\ e^{el}_{yy} \end{bmatrix} = \begin{bmatrix} \alpha & 0 & \beta \\ 0 & \omega & 0 \\ \beta & 0 & \gamma \end{bmatrix} \begin{bmatrix} e^{el}_{xx} \\ e^{el}_{xy} \\ e^{el}_{yy} \end{bmatrix} \tag{4}$$

Given the information, a forward model takes x_m, f and $\boldsymbol{C}$ as input and calculates the displacement vector $\boldsymbol{d}$ by solving a strain energy minimization problem: $\min_{\boldsymbol{d}} \frac{1}{2} \int \sigma e^{el} dV$, for which the KKT solution is the stress equilibrium condition, also known as stress continuity equation, with a pure Neumann boundary condition ($\hat{\boldsymbol{n}}$ is the unit surface norm vector):

$$\nabla \cdot \sigma = 0, \quad \hat{\boldsymbol{n}} \cdot \sigma = 0 \tag{5}$$

Define $c := \begin{bmatrix} \alpha & 0 & 0 & \beta \\ 0 & \frac{\omega}{2} & \frac{\omega}{2} & 0 \\ 0 & \frac{\omega}{2} & \frac{\omega}{2} & 0 \\ \beta & 0 & 0 & \gamma \end{bmatrix}$, a forward model solves for $\boldsymbol{d}$ by the following equation:

$$\nabla \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f), \quad \hat{\boldsymbol{n}} \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f) \tag{6}$$

2.2 Inverse Learning: PDE constrained optimization

With the advancement in microscopy, we can spatially determine displacement $\hat{\boldsymbol{d}}$ and x_m, and hope to learn $f(x_m)$ and c (namely $\alpha, \beta, \gamma, \omega$). We divide our effort into 2 minimization problems: (1) assuming we know c, learn f; (2) assuming f is known, learn c. The first task is reasonable since c is computationally cheap to obtain from first principle calculations. The second task uses f from (1) and calibrates c, which can be regarded as the posterior estimation of c from data. The two problems are set up as the following:

Task 1. composition-strain relation retrieval

$$\min_f \quad F := \|\hat{\boldsymbol{d}} - \boldsymbol{d}\|^2 \tag{7}$$

$$\text{s.t.} \quad \nabla \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f), \tag{8}$$

$$\hat{\boldsymbol{n}} \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f) \tag{9}$$

Task 2. stress-strain relation retrieval

$$\min_{\alpha,\beta,\gamma,\omega} \quad F := \|\hat{\boldsymbol{d}} - \boldsymbol{d}\|^2 \tag{10}$$

$$\text{s.t.} \quad \nabla \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f), \tag{11}$$

$$\hat{\boldsymbol{n}} \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f) \tag{12}$$

3 Methods and Algorithms

3.1 Finite Element Discretization

Finite element discretization allows for the relaxation of the PDE constraint into algebraic equality constraints. We employed a 3-node triangular element, with number of grid points to be tuned in the following experiment. Details of the process, including formulation, matrix assembly, and explicit expressions for each term can be found in Appendix Section 6. Note that in the notation below, all flattened column vectors are represented with a non-bold form for convenience. For example, for vector $\boldsymbol{d} \in \mathbb{R}^2$ at multiple locations, its total nodal representation is flattened into $d \in \mathbb{R}^{2N_p}$, where N_p is the number of nodes. Our two optimization tasks are thus the following:

Task 1. composition-strain relation retrieval

$$\min_{f} \quad F := \|\hat{d} - d\|^2 \tag{13}$$

$$\text{s.t.} \quad K(\alpha, \beta, \gamma, \omega)d = G[f] \tag{14}$$

Here, $K(\alpha, \beta, \gamma, \omega) \succeq 0$ is the assembled stiffness matrix that is known, and G is a column vector that's a linear transformation of f derivative values at each node.

Task 2. stress-strain relation retrieval

$$\min_{\alpha,\beta,\gamma,\omega} \quad F := \|\hat{d} - d\|^2 \tag{15}$$

$$\text{s.t.} \quad K(\alpha, \beta, \gamma, \omega)d = G \tag{16}$$

Here, $K(\alpha, \beta, \gamma, \omega) = \alpha A + \beta B + \gamma C + \omega D \succeq 0$, where A, B, C, D are given fixed matrices, and $G = \alpha g_1 + \beta g_2 + \gamma g_3 + \omega g_4$ where g_i is fixed and known.

3.2 Legendre Polynomial Basis Expansion

For task 1, we want to retrieve a function defined over $C[0, 1]$. Based on some physics intuition [3, 8], we approximate f by a finite Legendre Polynomial series with $f_x(x_m) = \sum_{i=1}^{N} a_N L_n(2x_m - 1)$, $f_y(x_m) = \sum_{i=1}^{N} b_N L_n(2x_m - 1)$ and keep N a hyper-parameter determined by data, to be discussed later. Using the method above, we can transform the right hand side of the algebraic constraint into $G[f] = Dp$, with $D = [D_1, D_2, ..., D_N, D_1, ..., D_N]$, $D_i = G[Ln(x_m)]$ and $p := (a_1; a_2; ...; a_N; b_1; b_2; ...; b_N)$ (see Appendix Section 6). We have

$$\min_{p} \quad F := \|\hat{d} - d\|^2 \tag{17}$$

$$\text{s.t.} \quad K(\alpha, \beta, \gamma, \omega)d = Dp \tag{18}$$

3.3 Data and Experiment

Data The data that we have are 2 images of size 200x100 of battery particle, shown in Figure 1, with x_m and $\boldsymbol{d} = (u, v)$. We divide the two into training and test image.

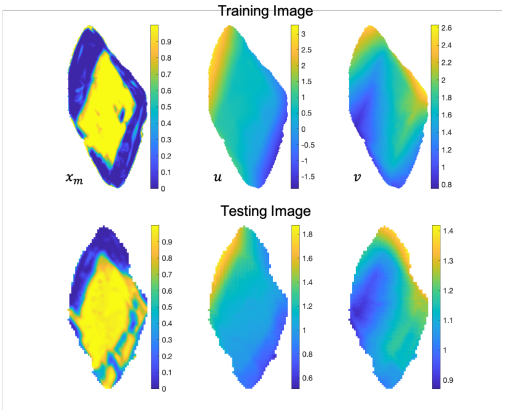


Figure 1: Experimental Data: 2 sets of correlated image dataset 200x100 pixels were provided. We divide the image data into training image and testing image for inverse learning.

Task 1. composition-strain retrieval Because the composition-strain relation is computationally challenging to obtain, the main effort has been on the first task. Furthermore, we showed in Appendix Section 7 that this is a unconstrained min-norm problem which is well studied in optimization theory, and we mostly focused on retrieving the correct physical model. As a benchmark, we first simulated data using the forward model with certain parameters, and then tested our algorithms to validate our inverse model, mesh size, convergence, and robustness of retrieval after adding Gaussian noise. All benchmark tests have been repeated for at least 100 times with a random initialization unless stated otherwise. The composition-strain relation was then retrieved from training image, and tested on test image to determine the cutoff degree number N in the Legendre Polynomial Expansion.

Task 2. stress-strain retrieval For completeness, we also explored the second task. Some basic conclusions on the optimal solution uniqueness and convexity were discussed in later sections. Inheriting knowledge from the first task, hyperparameters were kept the same as the first task. Since the problem is not convex, we focused our major attention on algorithmic aspect. Three different algorithms including steepest descent, conjugate gradient, and Newton's method were implemented and compared.

3.4 Algorithms

For task 1, the optimal solution was explicitly derived as: $p = D\backslash(K * \hat{d})$. For task 2, because the problem is no longer convex (see Section 4.2), algorithms to reach first order KKT conditions were explored, including steepest descent (SDM), conjugate gradient (CG) and Newton's method (NM). SDM with an adaptive stepsize using backtracking line search was initially explored and fine tuned in order to gain some insight into the solution structure. Derivatives were derived in Appendix Section 9 and verified via numerical differentiation.

4 Discussion

4.1 Composition-strain retrieval

For task 1, our goal is to obtain $p \in \mathbb{R}^{2N}$ by solving the following optimization problem: $\min_p \ F := \|\hat{d} - d\|^2, \ \text{s.t.} \ K(\alpha, \beta, \gamma, \omega)d = Dp$.

Optimal Solution Uniqueness $K \succeq 0$ because $\text{rank}(K) = 2N_p - 3$ for a pure Neumann boundary condition, due to the slackness in the translational and rotational 3 degrees of freedom [1]. Hence, the numerical condition number of K is near infinity, and thus makes inverse learning impossible using the explicit expressions derived (see Appendix Section 6). However, we can overcome this problem by fixing these degrees of freedom based on the experimental observation. After applying the 3 fixed-point constraint mentioned and deleting redundant algebraic equations, $\text{rank}(K) = 2N_p$ and so $K \succ 0$. In general p has a low cutoff number, therefore task 1 is an unconstrained minimum linear least-squares problem with a unique solution.

Mesh Size h_{max} We explored the effect of discretization on the recovery of p as our benchmark tests. As stated earlier, we tested our inverse recovery of p with 4 different mesh sizes, with randomized p different in both size and values. As shown in Table 1, a finer grid size improves the recovery accuracy, while higher cutoff number recovery of p is slightly less accurate. We therefore chose a h_{max} of 1, which has the highest accuracy without compromising computation time.

h_{max}	N=1	N = 2	N = 10
100	1.2e-14	2.4e-13	3.7e-9
10	4.9e-15	2.5e-14	2.0e-12
1	2.5e-14	3.7e-14	5.7e-13
0.1	8.9e-14	NC	NC

Table 1: L2 norm of $(p_{retrieved} - p_{true})$ as a function of h_{max} and cut-off number. h_{max} is based on pixel length. NC means not computed due to limited computation time.

Cutoff Number N We also explored the accuracy of finite cutoff number. We generated a random function f parameterized by p up to 10 degrees and used a lower cutoff number to retrieve. From Figure 2, we observe that as the degree number goes up, function approximation error, which is represented by the residual squared, approaches 0. This makes sense since the lower degree terms mostly capture lower frequency variations, which is the major trend.

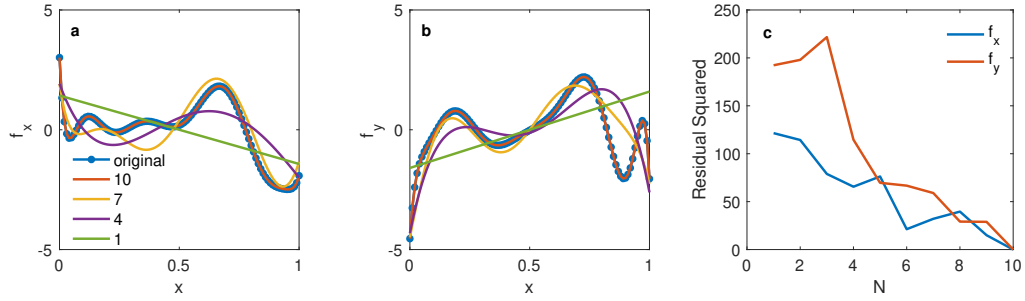


Figure 2: Finite N error on function $f_x(x_m), f_y(x_m)$ recovery. (a) and (b) show the function inversely recovered. (c) shows the error defined as the squared difference integral over $[0, 1]$.

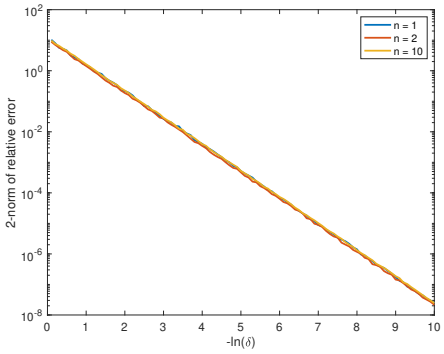


Figure 3: Recovery accuracy as a function of noise to signal ratio δ

Robustness Against Experimental Noise Another practical concern that needs to be tested is the experimental noise in our measurement data $\hat{d}$. We verified the robustness of this approach by adding artificial Gaussian noise on our observed $\hat{d}$ and evaluated the recovery of p. We varied noise to signal ratio (standard deviation δ) while keeping a zero-mean to the noise. As shown in Figure 3, recovery accuracy increases with a decreasing magnitude of noise with a power 0.9, independent of the cutoff number. The result indicates that the method is highly robust since our measurement noise is at best 7%, and therefore would yield a recovery accuracy of 3% in p.

Retrieval of composition-strain from experimental data Using the aforementioned approach, we extracted out the composition-strain relation from the experimental training image. Function f with different cutoff numbers were retrieved, as shown in Figure 4a and b. All curves retrieved follow the same major trend. As can be seen in Figure 4c, higher cutoff number captures more local fluctuations in the curve, with decreasing fitting error; however, probability of overfit exists. Validation from the test image indicates that the best cutoff number is 2, with the lowest test error shown in Figure 4d. The result is also physical as most theory predicts on first order a linear curve with some minor local fluctuations, which means that lower cutoff number is preferred [3]. Despite lack of direct measurement data on $f(x_m)$, other experiments have indicated that the change in f_x and f_y between x_m at 0 and 1 are 0.05 and -0.03 respectively, which is close to what we've extracted from experiment. To our best knowledge, this is the first time the composition-strain relation has been extracted for battery materials, which has been a long-standing challenge in the battery community [2, 6].

4.2 Stress-strain retrieval

For task 2, our goal is to obtain $(\alpha, \beta, \gamma, \omega) \in \mathbb{R}^4_+$ by solving the following problem:

$$\min_{\alpha,\beta,\gamma,\omega} \quad F := \|\hat{d} - d\|^2,$$

$$\text{s.t.} \quad Kd = G, \text{ where } K = \alpha A + \beta B + \gamma C + \omega D \succeq 0, \quad G = \alpha g_1 + \beta g_2 + \gamma g_3 + \omega g_4.$$

Existence of multiple optimal solutions First, we can easily see that multiple optimal solution exists. From Eqn 6, we can see that if $\{\alpha, \beta, \gamma, \omega\}$ is optimal, $\{k\alpha, k\beta, k\gamma, k\omega\}$ is optimal $\forall k \in \mathbb{R}$. This is because k can be canceled out from c on both sides. This implies that obtaining the physical parameters may be impractical. However, given the structure of the optimal solution sets, obtaining the relative ratio between the stiffness tensor constants is still possible once optimal is reached.

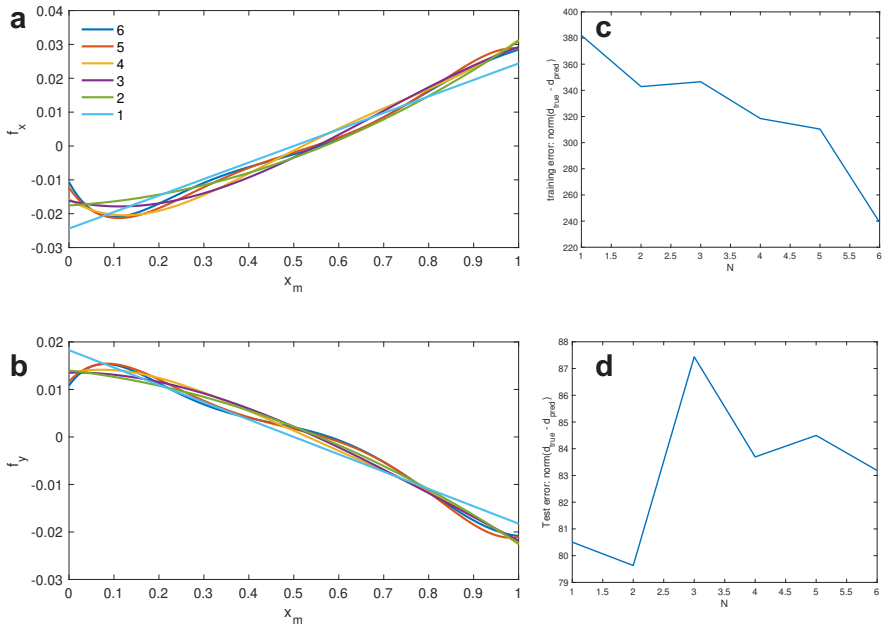
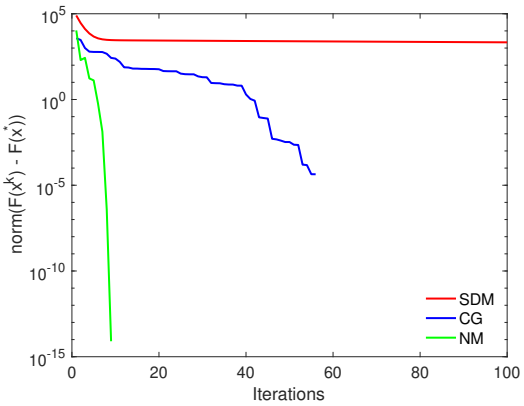


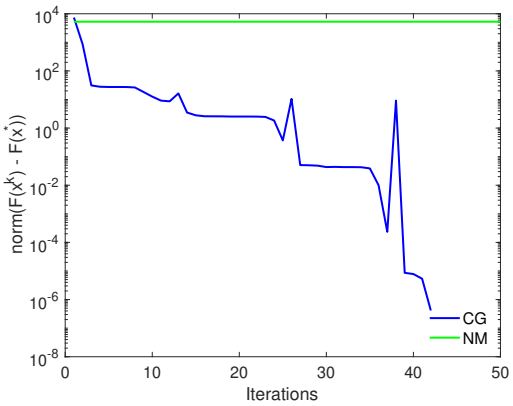
Figure 4: Composition-strain relation inversely learned from experimental data. (a),(b): Inversely learned f_x, f_y from training image with cutoff number 1 to 6. (c),(d): Training and test error as a function of N. As can be seen, N = 2 has the lowest test error, which is adopted.

Convexity Analysis We can show with an counter example that the minimization problem is non-convex (see Appendix Section 8.2). This makes sense as even in 1-D, the problem is non-convex. We therefore focused our attention on algorithms that reached first order KKT conditions.

Comparison of different algorithms We tested our algorithms in terms of speed and accuracy. We generated artificial data with a known parameter set and attempted to retrieve the parameter, similar to task 1. Because the algebraic constraint involves too many grid points that constitutes the matrix, computation is slow. We first benchmarked the algorithms on a constant force model by having f_{const} on the right-hand side (RHS) of the PDE in Eqn 6. Although the constant force model is of a different physical origin, it can run much faster due to vectorization and may shed light on our algorithms. The constant force model and the true model are similar for the following reasons: (1) stiffness matrix K is identical, and (2) the problem is non-convex (see Appendix Figure 6). Because true values of these parameters (stiffness tensors) often exist in literature, and are less computationally expensive to compute, we can start with theoretical values as our initial guesses. This means that the initial solution should be close to optimal. Similarly, we initialize within 10% of true values. Figure 5 a shows a comparison between SDM, CG and Newton's method for optimal $\alpha, \beta, \gamma, \omega$ recovery. Within our expectation, Newton performs the best, with CG being second, and SDM being last in terms of convergence speed. We hence compared performance of CG and Newton's method in the true model. We found that only CG yields an optimal solution within reasonable iterations whereas Newton's method fails to converge, as shown in Figure 5b. In fact, Newton's method won't converge due to the homogeneous structure of the objective function: Newton's method will generate an update direction where all parameters increase or decrease by the same ratio; this however will not work for our system since linearly changing all parameters with the same ratio will obtain the same objective function, and hence the algorithm will forever



(a) $||F(x^k) - F(x^*)||$



(b) $||F(x^k) - F(x^*)||$

Figure 5: (a): Comparison of Steepest Descent (SDM), Conjugate Gradient (CG) and Newton's method (NM) with initial values $\alpha_0, \beta_0, \gamma_0, \omega_0$ all within 10% of true values on the simplified constant force model. (b): CG and NM on the true system. Of note, NM failed to converge for the true system.

be stuck with the same objective value. CG on the other hand does not have such constraint and thus works for our minimization problem.

5 Conclusion

In conclusion, we've demonstrated a general data-driven inverse learning framework for extraction of constitutive relations by solving a PDE-constrained optimization problem. FEM was adopted to transform the PDE constraint into algebraic equality constraint, which made conventional optimization algorithms possible. The aforementioned framework was directly applied to battery materials with 2 sets of correlated images. Latent lithium composition-strain constitutive relation were successfully extracted for the first time. For completeness, extraction of the stiffness tensor from the stress-strain relation was also explored. Due to the non-uniqueness nature of optimal solution, we switched our attention to the relative ratio extraction between the parameters (namely stiffness anisotropy), which is nonetheless scientifically valuable. Comparison of the algorithms from multiple cases indicates that CG works best while Newton's method fails to reach optimum for the same task due to the solution structure of the problem.

References

[1] Pavel Bochev and Richard B Lehoucq. "On the finite element solution of the pure Neumann problem". In: *SIAM review* 47.1 (2005), pp. 50–66.

[2] Daniel A Cogswell and Martin Z Bazant. "Coherency strain and the kinetics of phase separation in LiFePO4 nanoparticles". In: *ACS nano* 6.3 (2012), pp. 2215–2225.

[3] Alan R Denton and Neil W Ashcroft. "Vegard's law". In: *Physical review A* 43.6 (1991), p. 3161.

[4] Henri P Gavin. *Mathematical properties of stiffness matrices*. 2006.

[5] Yiyang Li et al. "Fluid-enhanced surface diffusion controls intraparticle phase transformations". In: *Nature materials* 17.10 (2018), pp. 915–922.

[6] Jongwoo Lim et al. "Origin and hysteresis of lithium compositional spatiodynamics within battery primary particles". In: *Science* 353.6299 (2016), pp. 566–571.

[7] Samira Pakravan et al. "Solving inverse-PDE problems with physics-aware neural networks". In: *arXiv preprint arXiv:2001.03608* (2020).

[8] Arthur D Pelton and Christopher W Bale. "Legendre polynomial expansions of thermodynamic properties of binary solutions". In: *Metallurgical Transactions A* 17.6 (1986), pp. 1057–1063.

[9] Olek C Zienkiewicz, Robert L Taylor, and Jian Z Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2005.

Appendices

6 FEM discretization of the PDE

We partition the domain Ω into n elements $\Omega = \bigcup_{e=1}^{n_{\text{elem}}} \Omega^e$. By the Galerkin variational equation[1], the element stiffness matrix is defined as

$$K^e = \mathcal{C} \int_{\Omega^e} (B_d^e)^T (B_d^e) d\Omega^e \tag{19}$$

where

$$\mathcal{C} = \begin{bmatrix} \alpha & 0 & \beta \\ \beta & 0 & \omega \\ 0 & \gamma & 0 \end{bmatrix} \tag{20}$$

and N_d^e will be defined below using 3-node triangular element.

Next, we use the 3-node triangle method to do 2-D analysis. Each triangular element is assigned with local node numbers $\{1, 2, 3\}$. The element shape functions in terms of element natural coordinates (ξ, η) are

$$\begin{cases} N_1^e(\xi, \eta) = \xi \\ N_2^e(\xi, \eta) = \eta \\ N_3^e(\xi, \eta) = 1 - \xi - \eta \end{cases} \tag{21}$$

Let $N^e = [N_1^e \; N_2^e \; N_3^e]$. Consider the isoparametric map $\mathbf{x} : \triangle \to \Omega^e$ such that $x(\xi, \eta) = \sum_{b=1}^3 N_b^e(\xi, \eta) x_b^e = N^e \mathbf{x}^e$ and $y(\xi, \eta) = \sum_{b=1}^3 N_b^e(\xi, \eta) y_b^e = N^e \mathbf{y}^e$. Then by chain rule

$$\frac{\partial N_b^e}{\partial(\xi, \eta)} = \begin{bmatrix} \partial N_b^e/\partial \xi \\ \partial N_b^e/\partial \eta \end{bmatrix} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi \\ \partial x/\partial \eta & \partial y/\partial \eta \end{bmatrix} \begin{bmatrix} \partial N_b^e/\partial x \\ \partial N_b^e/\partial y \end{bmatrix} = (J^e)^T \frac{\partial N_b^e}{\partial(x, y)} \tag{22}$$

where we defined $B_b^e = \partial N_b^e / \partial(x, y)$.

The elements of Jacobian matrix is

$$(J^e)^T = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1^e & y_1^e \\ x_2^e & y_2^e \\ x_3^e & y_3^e \end{bmatrix} = \begin{bmatrix} x_1^e - x_3^e & y_1^e - y_3^e \\ x_2^e - x_3^e & y_2^e - y_3^e \end{bmatrix} \tag{23}$$

Denote the Jacobian determinant be

$$\det(J^e) = 2A^e \tag{24}$$

then the inverse Jacobian is

$$\begin{bmatrix} \partial N_b^e/\partial x \\ \partial N_b^e/\partial y \end{bmatrix} = \frac{1}{2A^e} \begin{bmatrix} y_2^e - y_3^e & -(y_1^e - y_3^e) \\ -(x_2^e - x_3^e) & x_1^e - x_3^e \end{bmatrix} \begin{bmatrix} \partial N_b^e/\partial \xi \\ \partial N_b^e/\partial \eta \end{bmatrix} \tag{25}$$

We get $\begin{cases} N_{1,1}^e = \partial N_1^e/\partial x = \frac{2}{A^e}(y_2^e - y_3^e) & N_{1,2}^e = \partial N_1^e/\partial y = \frac{2}{A^e}(x_3^e - x_2^e) \\ N_{2,1}^e = \partial N_2^e/\partial x = \frac{2}{A^e}(y_3^e - y_1^e) & N_{2,2}^e = \partial N_2^e/\partial y = \frac{2}{A^e}(x_1^e - x_3^e) \\ N_{3,1}^e = \partial N_3^e/\partial x = \frac{2}{A^e}(y_1^e - y_2^e) & N_{3,2}^e = \partial N_3^e/\partial y = \frac{2}{A^e}(x_2^e - x_1^e) \end{cases}$

[1] see Chapter 7 and 13 of the lecture notes for CME 232

So Eqn (19) can be written as

$$K^e = \int_{\Omega^e} B_d^e \ \mathcal{C} \ B_d^e \ d\Omega^e \tag{26}$$

$$= A^e \big[\frac{\partial N^e}{\partial(x^e, y^e)}\big]^T \ \mathcal{C} \ \frac{\partial N^e}{\partial(x^e, y^e)} \tag{27}$$

$$= \frac{1}{4A^e} \begin{bmatrix} y_3^e - y_1^e & 0 \\ 0 & x_3^e - x_2^e \\ x_3^e - x_2^e & y_2^e - y_3^e \end{bmatrix}^T \ \mathcal{C} \ \begin{bmatrix} y_3^e - y_1^e & 0 \\ 0 & x_3^e - x_2^e \\ x_3^e - x_2^e & y_2^e - y_3^e \end{bmatrix} \tag{28}$$

Note that for the matrix $B^e = \begin{bmatrix} y_3^e - y_1^e & 0 \\ 0 & x_3^e - x_2^e \\ x_3^e - x_2^e & y_2^e - y_3^e \end{bmatrix}$, each entry is a function of position and is independent of α, β, γ, and ω. So each entry of K^e is only at most a linear function of $\alpha, \beta, \gamma, \omega$ from $\mathcal{C}$.

Then using the direct stiffness method, the stiff matrix is $K = A_{e=1}^{n_{\text{elem}}} K^e$, which satisfies $Kd = G$, equivalent to Eqn (6).

Let $\nabla \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f) = G_\Omega$ and $\hat{\boldsymbol{n}} \cdot (c \otimes \nabla \boldsymbol{d}) = \nabla \cdot (c \otimes f) = G_{\partial\Omega}$ Then

$$G = \overset{n_{\text{elem}}}{\underset{e=1}{A}} G_{\partial\Omega}^e + \overset{n_{\text{elem}}}{\underset{e=1}{A}} G_\Omega^e \tag{29}$$

Note $G = G(x, y)$ is a function of position and suppose we parameterize at each position $G(x, y) = g(x_m) = \sum_{n=1}^{10} a_n L_n(x_m)$ (as in section 3.2).

Then

$$G_{\partial\Omega}^e = \int_{\partial\Omega^e} (N^e)^T \begin{bmatrix} g_1^e \\ g_2^e \end{bmatrix} d\partial\Omega^e \tag{30}$$

$$= \int_{\partial\Omega^e} (N^e)^T \begin{bmatrix} \sum_{n=1}^{10} a_n L_n(x_m) \\ \sum_{n=1}^{10} b_n L_n(x_m) \end{bmatrix} d\partial\Omega^e \tag{31}$$

$$= \sum_{n=1}^{10} \begin{bmatrix} a_n \\ b_n \end{bmatrix} \int_{\partial\Omega^e} (N^e)^T L_n(x_m) d\partial\Omega^e \tag{32}$$

Similarly

$$G_\Omega^e = \int_{\Omega^e} (N^e)^T \begin{bmatrix} g_1^e \\ g_2^e \end{bmatrix} d\Omega^e = \sum_{n=1}^{10} \begin{bmatrix} a_n \\ b_n \end{bmatrix} \int_{\Omega^e} (N^e)^T L_n(x_m) d\Omega^e \tag{33}$$

So

$$G = \overset{n_{\text{elem}}}{\underset{e=1}{A}} \sum_{n=1}^{10} \begin{bmatrix} a_n \\ b_n \end{bmatrix} \left(\int_{\partial\Omega^e} (N^e)^T L_n(x_m) d\partial\Omega^e + \int_{\Omega^e} (N^e)^T L_n(x_m) d\Omega^e \right) \tag{34}$$

$$= \sum_{n=1}^{N} \begin{bmatrix} a_n \\ b_n \end{bmatrix} \overset{n_{\text{elem}}}{\underset{e=1}{A}} \left(\int_{\partial\Omega^e} (N^e)^T L_n(x_m) d\partial\Omega^e + \int_{\Omega^e} (N^e)^T L_n(x_m) d\Omega^e \right) \tag{35}$$

$$= \sum_{n=1}^{N} \begin{bmatrix} a_n \\ b_n \end{bmatrix} D_n \tag{36}$$

$$= \begin{bmatrix} D_1 & ... & D_n & D_1 & ... & D_n \end{bmatrix} \begin{bmatrix} a_1 \\ ... \\ a_n \\ b_1 \\ ... \\ b_n \end{bmatrix} = Dp \tag{37}$$

Let $a = \begin{bmatrix} a_1 \\ ... \\ a_n \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ ... \\ b_n \end{bmatrix}$. Then

$$Kd = G \implies Kd = Dp \tag{38}$$

7 Derivation on Task 1: composition-strain relation minimization

The minimization is:

$$\min_{p \in \boldsymbol{R}^{2N}} \quad F := \|\hat{d} - d\|^2 \tag{39}$$

$$\text{s.t.} \quad K(\alpha, \beta, \gamma, \omega)d = Dp \tag{40}$$

When $K \succ 0$, $d = K^{-1}Dp$. The minimization then becomes:

$$\min_{p \in \boldsymbol{R}^{2N}} \quad F := \|\hat{d} - K^{-1}Dp\|^2 \tag{41}$$

The problem is thus an unconstrained least-squares problem, since the system is over-determined due to the number of grid points far greater than the unknowns. Thus the linear least squares solution for the problem is $D \backslash (Kd)$. Note: for proof of positive semi-definite properties of K, refer to [4] for details.

8 Convexity Analysis of Task 2: stress-strain relation minimization

For the following analysis, we set $\alpha_{true} = 2.5e3$, $\beta_{true} = 765$, $\gamma_{true} = 2.59e3$, $\omega_{true} = 858$ for the forward model and change $f(x_m)$ depending on the situation to generate $\hat{d}$ values.

8.1 Convex analysis of a test system: a constant force model

For computation simplicity, we set the RHS of Eqn 6 to be a constant, e.g. $[1; 0]$, we use simulated data and observe from plots that the problem is not convex. In Figure 6, we see the change of objective function value g on log scale when α and γ change. A line color that is closer to dark blue (red) means $\log(F)$ is small (large). We fix β be $\beta_{true} + 300$ and ω be $\omega_{true} + 300$. Lower left corner in the Figure 6 with multiple blue local minimum regions indicates that the problem is not convex.

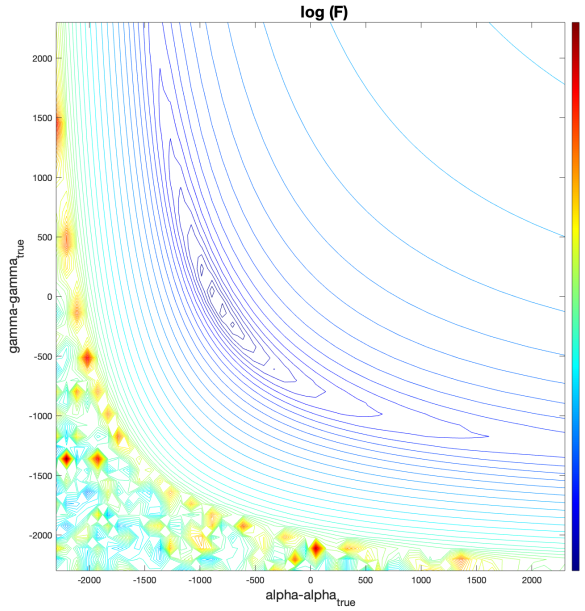


Figure 6: Take 50 values in range (-2300, 2300) as the difference of γ and γ_{true} and of α and α_{true}. Fix β and ω for $\beta = \beta_{true} + 300$ and $\omega = \omega_{true} + 300$. The colored contour plot shows $\log(F)$ in Eqn 15. Observe that in the bottom left corner a few regions with small $\log(F)$ are local minimum, which means that the problem is not convex.

8.2 Non-convexity of Task 2

Below, we provide a counter example that shows the non-convexity nature of the problem. Using the model that generates the artificial data with a known parameter set in section 4.2, we consider the following example:

Let two pairs of parameters $p(1), p(2)$ be $p(1) = \{\alpha = 62.2, \beta = 31.6, \gamma = 2589.7, \omega = 858\}$ and $p(2) = \{\alpha = 551.8, \beta = 276.3, \gamma = 2589.7, \omega = 858\}$. Then $F(p(1)) = 5.9978 * 10^6$ and $F(p(2)) = 4.8803 * 10^6$. Since $F(0.5 * p(1) + 0.5 * p(2)) = 5.8705 * 10^6$ and $0.5 * F(p(1)) + 0.5 * F(p(2)) = 5.439 * 10^6$, then $F(0.5 * p(1) + 0.5 * p(2)) > 0.5 * F(p(1)) + 0.5 * F(p(2))$, which violates the definition of convexity. Thus, Task 2 is non-convex when RHS of Eqn 6 is not constant.

9 Hessian and Jacobian for Task 2

We consider first when RHS of Eqn 6 is constant. We have:

$$\begin{aligned} \min_{\alpha,\beta,\gamma,\omega} \quad & F := \|\hat{d} - d\|^2 \\ & K(\alpha,\beta,\gamma,\omega)d = G \end{aligned} \tag{42}$$

The KKT conditions are

$$\begin{aligned} \frac{\partial F}{\partial \alpha} &= -2\hat{d}\frac{\partial d}{\partial \alpha} + 2d\frac{\partial d}{\partial \alpha} = 0 \\ \frac{\partial F}{\partial \beta} &= -2\hat{d}\frac{\partial d}{\partial \beta} + 2d\frac{\partial d}{\partial \beta} = 0 \\ \frac{\partial F}{\partial \gamma} &= -2\hat{d}\frac{\partial d}{\partial \gamma} + 2d\frac{\partial d}{\partial \gamma} = 0 \\ \frac{\partial F}{\partial \omega} &= -2\hat{d}\frac{\partial d}{\partial \omega} + 2d\frac{\partial d}{\partial \omega} = 0 \end{aligned} \tag{43}$$

When $f(x_m)$ is a constant, K is linear of $\alpha, \beta, \gamma, \omega$ and G is a constant, i.e., $K(\alpha,\beta,\gamma,\omega) = \alpha A + \beta B + \gamma C + \omega D + E$. When $f(x_m)$ is a Legendre polynomial, K remains the same, but G also becomes linear of $\alpha, \beta, \gamma, \omega$, i.e. $G(\alpha,\beta,\gamma,\omega) = \alpha g1 + \beta g2 + \gamma g3 + \omega g4 + g5$.

Note that

$$\begin{aligned} A = K(1,0,0,0), B = K(0,1,0,0), C = K(0,0,1,0), D = K(0,0,0,1), E = K(0,0,0,0) = 0 \\ g1 = G(1,0,0,0), g2 = G(0,1,0,0), g3 = G(0,0,1,0), g4 = G(0,0,0,1), g5 = G(0,0,0,0) = 0 \end{aligned} \tag{44}$$

When $f(x_m)$ is a constant, we have

$$\begin{aligned} \frac{\partial d}{\partial \alpha} &= \frac{\partial K^{-1}G}{\partial \alpha} = -K^{-1}AK^{-1}G \\ \frac{\partial d}{\partial \beta} &= \frac{\partial K^{-1}G}{\partial \beta} = -K^{-1}BK^{-1}G \\ \frac{\partial d}{\partial \gamma} &= \frac{\partial K^{-1}G}{\partial \gamma} = -K^{-1}CK^{-1}G \\ \frac{\partial d}{\partial \omega} &= \frac{\partial K^{-1}G}{\partial \omega} = -K^{-1}DK^{-1}G \end{aligned} \tag{45}$$

Since A, B, C, D are fixed, at k^{th} iteration, we have $K(\alpha^k, \beta^k, \gamma^k, \omega^k) = \alpha^k A + \beta^k B + \gamma^k C + \omega^k D$. Plugging this into Eqn (45) and further into Eqn (43) we can compute the gradient of objective function at each iteration, which help us update our parameters $\alpha, \beta, \gamma, \omega$ as most of the optimization algorithms require the gradient of objective function as their search direction.

When $f(x_m)$ is a Legendre polynomial, we have

$$\begin{aligned} \frac{\partial d}{\partial \alpha} &= \frac{\partial K^{-1}G}{\partial \alpha} = -K^{-1}AK^{-1}G + K^{-1}g1 \\ \frac{\partial d}{\partial \beta} &= \frac{\partial K^{-1}G}{\partial \beta} = -K^{-1}BK^{-1}G + K^{-1}g2 \\ \frac{\partial d}{\partial \gamma} &= \frac{\partial K^{-1}G}{\partial \gamma} = -K^{-1}CK^{-1}G + K^{-1}g3 \\ \frac{\partial d}{\partial \omega} &= \frac{\partial K^{-1}G}{\partial \omega} = -K^{-1}DK^{-1}G + K^{-1}g4 \end{aligned} \tag{46}$$

At k^{th} iteration, we have $K(\alpha^k, \beta^k, \gamma^k, \omega^k) = \alpha^k A + \beta^k B + \gamma^k C + \omega^k D$ and $G(\alpha^k, \beta^k, \gamma^k, \omega^k) = \alpha^k g1 + \beta^k g2 + \gamma^k g3 + \omega^k g4$. Plugging this into Eqn (46) and further into Eqn (43) we can compute the gradient of objective function at each iteration for varying f.

We can further compute the Hessian matrix for our objective function (42) as follows for the purpose of Newton's method:
When $f(x_m)$ is a constant, we have

$$\begin{aligned}
\frac{\partial^2 d}{\partial \alpha^2} &= \frac{\partial^2 K^{-1}G}{\partial \alpha^2} = 2K^{-1}AK^{-1}AK^{-1}G \\
\frac{\partial^2 d}{\partial \beta^2} &= \frac{\partial^2 K^{-1}G}{\partial \beta^2} = 2K^{-1}BK^{-1}BK^{-1}G \\
\frac{\partial^2 d}{\partial \gamma^2} &= \frac{\partial^2 K^{-1}G}{\partial^2 \gamma^2} = 2K^{-1}CK^{-1}CK^{-1}G \\
\frac{\partial^2 d}{\partial \omega^2} &= \frac{\partial^2 K^{-1}G}{\partial \omega^2} = 2K^{-1}DK^{-1}DK^{-1}G \\
\frac{\partial^2 d}{\partial \alpha \partial \beta} &= \frac{\partial^2 K^{-1}G}{\partial \alpha \partial \beta} = K^{-1}AK^{-1}BK^{-1}G + K^{-1}BK^{-1}AK^{-1}G \\
\frac{\partial^2 d}{\partial \alpha \partial \gamma} &= \frac{\partial^2 K^{-1}G}{\partial \alpha \partial \gamma} = K^{-1}AK^{-1}CK^{-1}G + K^{-1}CK^{-1}AK^{-1}G \\
\frac{\partial^2 d}{\partial \alpha \partial \omega} &= \frac{\partial^2 K^{-1}G}{\partial \alpha \partial \omega} = K^{-1}AK^{-1}DK^{-1}G + K^{-1}DK^{-1}AK^{-1}G \\
\frac{\partial^2 d}{\partial \beta \partial \gamma} &= \frac{\partial^2 K^{-1}G}{\partial \beta \partial \gamma} = K^{-1}BK^{-1}CK^{-1}G + K^{-1}CK^{-1}BK^{-1}G \\
\frac{\partial^2 d}{\partial \beta \partial \omega} &= \frac{\partial^2 K^{-1}G}{\partial \beta \partial \omega} = K^{-1}BK^{-1}DK^{-1}G + K^{-1}DK^{-1}BK^{-1}G \\
\frac{\partial^2 d}{\partial \gamma \partial \omega} &= \frac{\partial^2 K^{-1}G}{\partial \gamma \partial \omega} = K^{-1}CK^{-1}DK^{-1}G + K^{-1}DK^{-1}CK^{-1}G
\end{aligned} \tag{47}$$

When $f(x_m)$ is a Legendre polynomial, we have

$$\begin{aligned}
\frac{\partial^2 d}{\partial \alpha^2} &= \frac{\partial^2 K^{-1}G}{\partial \alpha^2} = 2K^{-1}AK^{-1}AK^{-1}G - 2K^{-1}AK^{-1}g1 \\
\frac{\partial^2 d}{\partial \beta^2} &= \frac{\partial^2 K^{-1}G}{\partial \beta^2} = 2K^{-1}BK^{-1}BK^{-1}G - 2K^{-1}AK^{-1}g2 \\
&\vdots \\
\frac{\partial^2 d}{\partial \alpha \partial \beta} &= \frac{\partial^2 K^{-1}G}{\partial \alpha \partial \beta} = K^{-1}AK^{-1}BK^{-1}G - K^{-1}AK^{-1}g2 + K^{-1}BK^{-1}AK^{-1}G - K^{-1}BK^{-1}g1 \\
&\vdots \\
\frac{\partial^2 d}{\partial \gamma \partial \omega} &= \frac{\partial^2 K^{-1}G}{\partial \gamma \partial \omega} = K^{-1}CK^{-1}DK^{-1}G - K^{-1}CK^{-1}g4 + K^{-1}DK^{-1}CK^{-1}G - K^{-1}DK^{-1}g3
\end{aligned}$$

Also note that given Eqn (43), we have

$$\frac{\partial^2 F}{\partial \alpha^2} = -2\hat{d}\frac{\partial^2 d}{\partial \alpha^2} + 2\frac{\partial^2 d}{\partial \alpha^2} + 2d\frac{\partial^2 F}{\partial \alpha^2}$$
$$\frac{\partial^2 F}{\partial \alpha \partial \beta} = -2\hat{d}\frac{\partial^2 d}{\partial \alpha \partial \beta} + 2\frac{\partial d}{\partial \beta}\frac{\partial d}{\partial \alpha} + 2d\frac{\partial^2 d}{\partial \alpha \partial \beta} \tag{48}$$

Similarly, we can obtain the second derivative of our objective function (42) for β, γ, ω. The Hessian matrix of our objective function g would be as follows:

$$\begin{pmatrix} \frac{\partial^2 F}{\partial \alpha^2} & \frac{\partial^2 F}{\partial \alpha \partial \beta} & \frac{\partial^2 F}{\partial \alpha \partial \gamma} & \frac{\partial^2 F}{\partial \alpha \partial \omega} \\ \frac{\partial^2 F}{\partial \alpha \partial \beta} & \frac{\partial^2 F}{\partial \beta^2} & \frac{\partial^2 F}{\partial \beta \partial \gamma} & \frac{\partial^2 F}{\partial \beta \partial \omega} \\ \frac{\partial^2 F}{\partial \alpha \partial \gamma} & \frac{\partial^2 F}{\partial \beta \partial \gamma} & \frac{\partial^2 F}{\partial \gamma^2} & \frac{\partial^2 F}{\partial \gamma \partial \omega} \\ \frac{\partial^2 F}{\partial \alpha \partial \omega} & \frac{\partial^2 F}{\partial \beta \partial \omega} & \frac{\partial^2 F}{\partial \gamma \partial \omega} & \frac{\partial^2 F}{\partial \omega^2} \end{pmatrix}$$

Once we obtained the gradient and hessian matrix of our objective function F, we chose and implemented three optimization algorithms to learn the true values of $\alpha, \beta, \gamma, \omega$, given $\hat{d}$ and $f(x_m)$. These three algorithms are shown as follows,

- First order method: Steepest Descent
- 1.5 order method: Conjugate Gradient
- Second order method: Newton's Method