Mathematical Formulation of Nelson-Siegel Model Calibration via Loss-Based Optimization

May 29, 2025

Student:

Hayoung Lee: hayoung.lee-2@student.uts.edu.au
Quoc Thai Tran: quocthai.tran@student.uts.edu.au
Stephanie Zhen: liyee.zhen@student.uts.edu.au
Ziqi Zhou: ziqi.zhou-3@student.uts.edu.au

Lecturer:

Hanyu Gu: hanyu.gu@uts.edu.au

School of Mathematical and Physical Sciences
Faculty of Science
University of Technology Sydney

Contents

1 Introduction

The accurate estimation of the yield curve is a cornerstone of fixed income analysis, crucial for pricing bonds, managing interest rate risk, and performing monetary policy analysis. Among the various term-structure models, the Nelson and Siegel (1987) (NS model) has become one of the most widely used due to its parsimonious structure, intuitive interpretation, and empirical performance.

Traditionally, the Nelson–Siegel curve is estimated by minimising the squared error between observed market yields and the model's fitted yields. However, in practice, each quoted yield used for error minimisation is typically taken as the midpoint of a bid-ask spread which varies significantly across bonds. The width of this spread contains valuable information about the bond's liquidity, uncertainty, and transaction costs. As a result, equally weighting all mid quotes can allow illiquid bonds with wide bid-ask spreads to disproportionately influence and distort the fitted curve, which can mislead interpretations.

To address this issue, a flexible calibration framework is proposed in order to embed bid–ask information directly during curve fitting. Specifically, a hinge-type loss function is designed such that it penalizes any fitted yield that falls outside the observed bid–ask interval with a quadratic cost, while imposing no penalty for in-spread fits. This retains the contribution of well-priced mid-quotes while substantially reduces any over-fitting caused any wide or illiquid spreads.

This project formally develops the mathematical formulation of the loss-based calibration approach, implements several gradient-based and derivative-free optimisers (benchmarked against scipy.optimize, the industry standard for optimisation in Python), and evaluates performance on Reserve Bank of Australia benchmark yield data. The ultimate goal is to develop a robust, interpretable calibration method that respects market-implied price bounds and enhances the reliability of term-structure estimation.

2 Literature Review

The Nelson–Siegel (NS) model was originally proposed by Nelson and Siegel (1987) as a parsimonious three-factor representation of the yield curve. Its ability to capture the level, slope and curvature of the term structure with just four parameters has led to widespread adoption by central banks and market participants.

Recognising that bond prices are typically quoted in the market with bid–ask spreads, Gomes-Gonçalves *et al.* (2017) calibrated short-rate term structure models using intervals rather than point prices. By treating every coupon-bond quotation as the admissible range $[\text{bid}, \text{ask}]$, it was shown that ignoring spreads systematically biases the estimated NS factors, especially for illiquid maturities. Hence, a maximum-entropy procedure was proposed to find the smoothest zero-coupon curve consistent with all intervals.

Based on this insight, Lapshin and Sohatskaya (2020) developed an optimisation scheme in which model prices are penalised *only when they fall outside* the observed bid–ask band. Several weighting and loss-function designs were experimented on and demonstrated using sovereign bond data, it was discovered that interval-aware calibration is capable of improving out-of-sample pricing accuracy and hedging stability without sacrificing parsimony.

Together, these studies underscored two key lessons: (i) bid–ask information contains valuable constraints that should be respected during curve fitting, and (ii) loss-based optimisation that tolerates any price within the trading range yields more robust NS parameter estimates. These ideas form the conceptual backbone of the calibration framework advanced in the present report.

3 Objectives & Scope

3.1 Objectives

The primary objective of this project is to calibrate the NS yield-curve model using a loss function that incorporates observed bid–ask spreads of coupon-bearing bonds. Specifically, we aim to:

1. Formulate a mathematically consistent loss function that penalises model-implied prices lying outside market bid–ask bounds
2. Derive the analytical expressions for bond pricing, discount factors and the gradient of the loss function with respect to model parameters
3. Implement and compare multiple optimisation algorithms
4. Analyse the numerical performance, convergence behaviour and robustness of each method

3.2 Scope

This study is limited to the calibration of the NS model on a cross section of fixed coupon, default free government bonds. The scope is defined by the following assumptions and boundaries:

1. Interest rates are modeled under a deterministic framework using continuously compounded yields
2. The model ignores liquidity premiums, tax considerations and default risk
3. Only static parameter estimation is considered—dynamic models or real-time updating are beyond this project's scope
4. The optimisation allows $\gamma \geq 0$.

4 Methodology

4.1 Mathematical Formulation and Loss-Based Calibration Framework

4.1.1 Nelson-Siegel Yield Curve Model

The instantaneous forward rate function is:

$$f(t) = f_0 + f_1 e^{-t/\gamma} + f_2 \cdot \frac{t}{\gamma} e^{-t/\gamma}$$

where the parameter vector is:

$$\boldsymbol{\theta} = (f_0, f_1, f_2, \gamma)^\top \in \mathbb{R}^4, \quad \text{with } \gamma \geq 0$$

In this model, f_0 represents the long-term level of the yield curve, f_1 controls the short-term component (slope), f_2 captures the medium-term component (curvature), and γ is the decay factor that determines the exponential shape of the curve.

4.1.2 Zero-Coupon Discount Factor

The forward rate is defined as:

$$f(\boldsymbol{\theta}, t) = f_0 + f_1 e^{-\frac{t}{\gamma}} + f_2 \cdot \frac{t}{\gamma} e^{-\frac{t}{\gamma}}$$

Then, the discount factor is:

$$B(\boldsymbol{\theta}, T) = \exp\left(-\int_0^T f(\boldsymbol{\theta}, t)\, dt\right)$$

Then,

$$\int_0^T f(\boldsymbol{\theta}, t)\, dt = \int_0^T \left(f_0 + f_1 e^{-\frac{t}{\gamma}} + f_2 \cdot \frac{t}{\gamma} e^{-\frac{t}{\gamma}}\right) dt = T f_0 + \left(\gamma - e^{-\frac{T}{\gamma}}\gamma\right) f_1 + \left(\gamma - e^{-\frac{T}{\gamma}}(T+\gamma)\right) f_2$$

Hence,

$$B(\boldsymbol{\theta}, T) = \exp\left[-T f_0 - \left(\gamma - e^{-\frac{T}{\gamma}}\gamma\right) f_1 - \left(\gamma - e^{-\frac{T}{\gamma}}(T+\gamma)\right) f_2\right]$$

4.1.3 Bond Pricing with Coupons

The price of bond j with n_j coupon payments is given by:

$$V_j = 100 \left(\sum_{i=1}^{n_j} c_j \cdot B(\boldsymbol{\theta}, T_i) + B(\boldsymbol{\theta}, T_{n_j}) \right)$$

4.1.4 Market Bid-Ask Bound Constraints

Let the observed market bid and ask prices of bond j be:

$$\text{Bid}_j, \quad \text{Ask}_j, \quad \text{with Bid}_j \leq \text{Ask}_j$$

We aim for $V_j(\theta) \in [\text{Bid}_j, \text{Ask}_j]$ to reflect realistic market pricing.

4.1.5 Loss Function Construction

We define a penalty-based loss function for bond j as:

$$\ell_j(\theta) = \left(\frac{\max(0, \text{Bid}_j - V_j(\theta))}{\text{Bid}_j} \right)^2 + \left(\frac{\max(0, V_j(\theta) - \text{Ask}_j)}{\text{Ask}_j} \right)^2$$

This function penalizes the model only when $V_j(\theta) \notin [\text{Bid}_j, \text{Ask}_j]$.

4.1.6 Total Loss Function

The total loss across all m bonds is:

$$L(\theta) = \sum_{j=1}^{m} \ell_j(\theta)$$

4.1.7 Optimization Objective

The loss function is continuous and differentiable almost everywhere with respect to the parameter vector $\boldsymbol{\theta} = (f_0, f_1, f_2, \gamma)^\top$ with the constraint $\gamma \geq 0$; expressions are evaluated for $\gamma > 0$ and extended by continuity at $\gamma = 0$.

We solve the following optimization problem:

$$\min_{f_0, f_1, f_2, \gamma} \left(\sum_{j=1}^{m} \left[\left(\frac{\max(0, \text{Bid}_j - V_j)}{Bid_j} \right)^2 + \left(\frac{\max(0, V_j - \text{Ask}_j)}{\text{Ask}_j} \right)^2 \right] \right)$$

4.1.8 Gradient and Hessian Calculation

Analytical

The first and second-order derivatives of the hinge-type loss function were derived analytically and implemented manually in code. It is continuously differentiable C^1, but not twice continuously differentiable C^2 (the second derivative exists piece-wise but is discontinuous at the bid-ask boundaries, although its behaviour can be approximated in the numerical section below). This level of smoothness is sufficient for standard gradient-based optimisation algorithms, although it involves inequality constraints and non-linearity due to the exponential structure of the NS model. Detailed derivations can be found in Appendix A.

Numerical

Choose a small step $\varepsilon > 0$ and let $\mathbf{e}_k \in \mathbb{R}^4$ denote the k-th standard basis vector. For the total loss

$$L(\boldsymbol{\theta}) = \sum_{j=1}^{m} \Big(\frac{\max\big(0, V_j(\boldsymbol{\theta}) - A_j\big)}{A_j} \Big)^2 \; + \; \Big(\frac{\max\big(0, B_j - V_j(\boldsymbol{\theta})\big)}{B_j} \Big)^2 . \; B_j = Bid_j \; , \; A_j = Ask_j$$

the k-th partial derivative is numerically approximated by the central-difference formula

$$\frac{\partial L}{\partial \theta_k}(\boldsymbol{\theta}) \;\approx\; \frac{L(\boldsymbol{\theta} + \varepsilon \mathbf{e}_k) \;-\; L(\boldsymbol{\theta} - \varepsilon \mathbf{e}_k)}{2\,\varepsilon}.$$

Defining the perturbed vectors $\boldsymbol{\theta}_k^{(+)} = \boldsymbol{\theta} + \varepsilon \mathbf{e}_k$ and $\boldsymbol{\theta}_k^{(-)} = \boldsymbol{\theta} - \varepsilon \mathbf{e}_k$, this may be rewritten component-wise as

$$\frac{\partial L}{\partial \theta_k}(\boldsymbol{\theta}) \;\approx\; \sum_{j=1}^{m} \frac{L_j\big(\boldsymbol{\theta}_k^{(+)}\big) - L_j\big(\boldsymbol{\theta}_k^{(-)}\big)}{2\,\varepsilon},$$

If one prefers to approximate the derivative of each bond value first, the same scheme yields

$$\frac{\partial V_j}{\partial \theta_k}(\boldsymbol{\theta}) \;\approx\; \frac{V_j(\boldsymbol{\theta}_k^{(+)}) - V_j(\boldsymbol{\theta}_k^{(-)})}{2\,\varepsilon}, \qquad \frac{\partial B(\boldsymbol{\theta}, T)}{\partial \theta_k} \;\approx\; \frac{B(\boldsymbol{\theta}_k^{(+)}, T) - B(\boldsymbol{\theta}_k^{(-)}, T)}{2\,\varepsilon}.$$

Denote by $\mathbf{e}_k, \mathbf{e}_\ell \in \mathbb{R}^4$ the standard basis vectors indexed by the two integers $k, \ell \in$

$\{0, 1, 2, 3\}$ ($k \le \ell$ after sorting). Define the four perturbed parameter vectors

$$\boldsymbol{\theta}^{(++)} = \boldsymbol{\theta} + \varepsilon \mathbf{e}_k + \varepsilon \mathbf{e}_\ell,$$
$$\boldsymbol{\theta}^{(+-)} = \boldsymbol{\theta} + \varepsilon \mathbf{e}_k - \varepsilon \mathbf{e}_\ell,$$
$$\boldsymbol{\theta}^{(-+)} = \boldsymbol{\theta} - \varepsilon \mathbf{e}_k + \varepsilon \mathbf{e}_\ell,$$
$$\boldsymbol{\theta}^{(--)} = \boldsymbol{\theta} - \varepsilon \mathbf{e}_k - \varepsilon \mathbf{e}_\ell,$$

and evaluate the loss at these points: $L^{(++)} = L(\boldsymbol{\theta}^{(++)})$, $L^{(+-)} = L(\boldsymbol{\theta}^{(+-)})$, $L^{(-+)} = L(\boldsymbol{\theta}^{(-+)})$, $L^{(--)} = L(\boldsymbol{\theta}^{(--)})$.

The mixed second-order partial derivative is then approximated by the central finite-difference formula

$$\frac{\partial^2 L}{\partial \theta_k \partial \theta_\ell}(\boldsymbol{\theta}) \;\approx\; \frac{L^{(++)} - L^{(+-)} - L^{(-+)} + L^{(--)}}{4\varepsilon^2}$$

.

4.2 Property of the loss function

The penalised objective is:

$$L(\boldsymbol{\theta}) = \sum_{j=1}^{m} \Big[\tfrac{\max(0, \mathrm{B}_j - V_j(\boldsymbol{\theta}))}{\mathrm{B}_j}\Big]^2 + \Big[\tfrac{\max(0, V_j(\boldsymbol{\theta}) - \mathrm{A}_j)}{\mathrm{A}_j}\Big]^2, \qquad \boldsymbol{\theta} = (f_0, f_1, f_2, \gamma)^\top,$$

where each model price $V_j(\boldsymbol{\theta}) = 100\big[c_j \sum_i B(\boldsymbol{\theta}, T_{ji}) + B(\boldsymbol{\theta}, T_{jn_j})\big]$ is non-linear in $\boldsymbol{\theta}$ through the discount factor

$$B(\boldsymbol{\theta}, T) = \exp\Big\{-T f_0 - (\gamma - e^{-T/\gamma}\gamma) f_1 - \big[\gamma - e^{-T/\gamma}(T + \gamma)\big] f_2\Big\}, \qquad \gamma \ge 0.$$

4.2.1 Differentiability of L

As shown in §4.1.8, the penalised loss $L(\theta)$ is continuously differentiable (C^1); its Hessian exists piece-wise but is discontinuous at the bid–ask boundaries. Hence analytic expressions (or stable finite–difference approximations) are available for both the gradient and the Hessian. This level of smoothness is sufficient for the use of first-order methods (steepest-descent, conjugate-gradient) as well as Newton-type or quasi-Newton algorithms (e.g. BFGS, L-BFGS-B, trust-region Newton) that can accommodate a piece-wise continuous Hessian.

4.2.2 Non-convexity of L

A twice–differentiable function is strictly convex only if its Hessian is positive-definite everywhere. To test this condition empirically, the eigenvalues of $\nabla^2 L(\theta)$ were evaluated on a four–dimensional grid

$$f_0, f_1, f_2 \in [-5, 5], \qquad \gamma \in (0, 20],$$

which covers the parameter region of practical interest for our model. For each grid point the Hessian was assembled analytically and its eigenvalues computed with NumPy's `eigvalsh`. Throughout the grid, the Hessian has always yielded non positive eigenvalues, failing to be positive-definite. Therefore L is non-convex on the domain explored, and any solution returned by an optimiser can only be regarded as a local minimum.

4.2.3 Optimisation Experiments

The loss function was first minimised with `scipy.optimize` using the quasi-Newton BFGS algorithm, chosen for its robustness on mildly non-convex problems. Through experimentation, however, BFGS was sensitive to the starting point: several initial guesses failed to converge or terminated at implausible local minima. To concentrate on the model relevant region of the parameter space, the initial vector was fixed at

$$x_0 = \begin{bmatrix} 0.042 \\ -0.0318 \\ -0.0268 \\ 1.72 \end{bmatrix},$$

matching the midpoint in Diebold and Li (2006). With this starting value, the optimiser converged to the local solution

$$x^* = \begin{bmatrix} 0.0644 \\ -0.0291 \\ 1.28 \times 10^{-5} \\ 17.97 \end{bmatrix}.$$

1. Two-dimensional loss surface

Figure 1 visualises the loss landscape in the $(f_0,\ f_1)$ plane while the remaining parameters are held fixed at their previously-optimised values $\big(f_2 = 1.28 \times 10^{-5},\ \gamma = 17.97\big)$. The background shading represents the normalised loss $L(f_0, f_1)$: dark purple indicates low values, whereas green/yellow denotes high values.

The red line shows the trajectory taken by the BFGS optimiser, which starts at the upper-right corner $[\,0.10,\ 0.10\,]$ and, after a sharp curvature-driven turn, descends into the darker basin. Convergence as shown is reached at point

$$x^* = [\,0.0644,\ -0.0291\,],$$

where the stopping criteria are satisfied. The visible turns in the path shows the successive updates of the BFGS inverse-Hessian approximation as it adapts to the locally varying curvature of the non-convex surface. The contrast between flat plateaus and steep ridges on the loss surface explains why the optimiser is so sensitive to its starting point, as observed in earlier experiments.

2. Three-dimensional loss surface

Figure 2 displays a three-dimensional slice of the loss surface, evaluated on the cube

$$f_0 \in [-0.06,\ 0.10], \qquad f_1 \in [-0.10,\ 0.10], \qquad f_2 \in [-0.10,\ 0.10],$$

with γ fixed at its previously optimised value.

The cloud reveals a narrow, almost linear valley of dark-coloured points running diagonally through the cube. Moving along this valley leaves the objective nearly unchanged, but even a small step in a transverse direction raises the loss sharply. This "canyon" landscape explain the reasons behind why the gradient-based algorithms sensitive to their initial guess and prone to large, kink-like turns (as seen in the two-dimensional path). In short, the 3-D view confirms why different starts can slide the optimiser along this flat ridge before it finally settles in a local minimum.

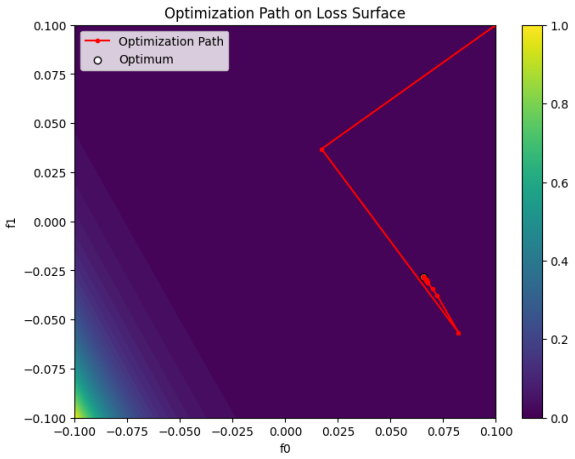

Figure 1: Optimisation path on the f_0–f_1 loss surface

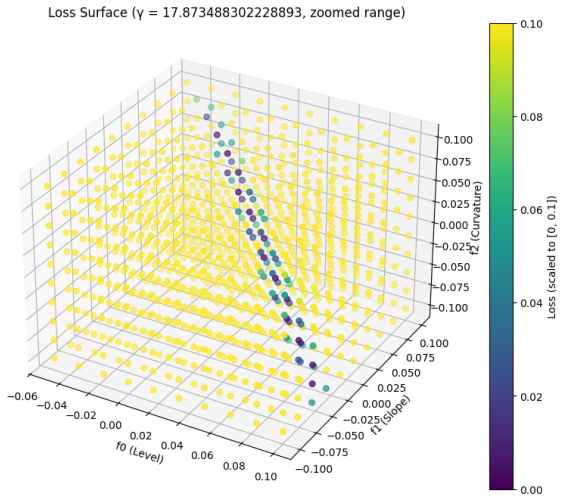

Figure 2: Optimisation path on the 3-D loss surface

4.3 Optimization Algorithms

Let

$$\boldsymbol{\theta} = \big(f_0, f_1, f_2, \gamma\big)^\top \in \mathbb{R}^3 \times [0, \infty),$$

and denote by $V_j(\boldsymbol{\theta})$ the model price of bond $j = 1, \ldots, m$ (see §4.1.3). For observed bid–ask bounds $\big[\mathrm{Bid}_j, \mathrm{Ask}_j\big]$, define the penalised loss

$$L(\boldsymbol{\theta}) = \sum_{j=1}^{m} \ell_j(\boldsymbol{\theta}), \quad \ell_j(\boldsymbol{\theta}) = \left[\frac{\max\big(0, \mathrm{Bid}_j - V_j(\boldsymbol{\theta})\big)}{\mathrm{Bid}_j}\right]^2 + \left[\frac{\max\big(0, V_j(\boldsymbol{\theta}) - \mathrm{Ask}_j\big)}{\mathrm{Ask}_j}\right]^2.$$

Let

$$\boldsymbol{g}(\boldsymbol{\theta}) = \nabla L(\boldsymbol{\theta}), \qquad H(\boldsymbol{\theta}) = \nabla^2 L(\boldsymbol{\theta}),$$

with closed-form expressions given in §4. The calibration proceeds through two deterministic iterative schemes, both driven solely by the triplet $(L, \boldsymbol{g}, H)$.

In this project, two approaches are taken: one manually coded from scratch, and the other using SciPy's built-in optimizer.

4.3.1 Constraint on γ

To enforce $\gamma \geq 0$ and avoid numerical overflow when penalty is active, two strategies are used:

1. Reparameterisation: set $\gamma = e^x$ with $x \in \mathbb{R}$. This renders $\gamma > 0$ automatically and allows use of unconstrained methods (e.g. DFP or conjugate-gradient on x).

2. Explicit constraint: retain $\gamma \geq 0$ and employ constrained optimisers (e.g. gradient-projection or Zoutendijk's method).

Under both approaches, the zero-coupon discount factor is defined piecewise by continuity:

$$B(\boldsymbol{\theta}, T) = \begin{cases} e^{-Tf_0}, & \gamma = 0, \\ \exp\Big\{-Tf_0 - \big[\gamma - \gamma e^{-T/\gamma}\big] f_1 - \big[\gamma - (T+\gamma)e^{-T/\gamma}\big] f_2\Big\}, & \gamma > 0. \end{cases}$$

The branch at $\gamma = 0$ follows by taking the limit $\gamma \to 0^+$, since for any fixed $t > 0$, as $\gamma \to 0^+$:

$$e^{-t/\gamma} = o(1) \to 0, \qquad \frac{t}{\gamma}\, e^{-t/\gamma} = o(1) \to 0.$$

Consequently, the exponential term and its first derivative both vanish at the boundary $\gamma = 0$, so the limit-extension of the loss is well-defined and continuous.

4.3.2 Manual Implementation Methods

To improve the speed of our manual optimization, SciPy's L-BFGS-B is employed solely for line-search, while the core update formulas were coded from scratch.

1. DFP (Davidon–Fletcher–Powell) Method The DFP method is a quasi-Newton algorithm designed for unconstrained optimization. It assumes the objective function is continuously differentiable and builds an approximation to the inverse Hessian matrix using gradient information only. At each iteration, the search direction is computed as a product of the inverse Hessian approximation and the negative gradient. A line search is used to determine the step size.

2. Fletcher–Reeves Method (Conjugate Gradient) The Conjugate Gradient (CG) method is suitable for large-scale unconstrained problems. Instead of following the steepest descent direction, CG computes mutually conjugate directions that allow faster convergence. The method avoids storing matrices explicitly, making it efficient in memory.

3. Zoutendijk Method Zoutendijk's method provides a theoretical framework for the convergence of constrained optimization algorithms that use descent directions and line search. The approach assumes the objective function is differentiable and that the step sizes satisfy conditions such as the Armijo rule. It ensures convergence to a stationary point by requiring that each descent direction forms a sufficiently small angle with the gradient.

4. Gradient Projection The Gradient Projection Method is explicitly designed for constrained optimization where the feasible region is convex. At each iteration, it takes a step along the negative gradient and then projects the point back onto the feasible set to maintain feasibility. This method assumes that the objective function is continuously differentiable and the projection can be efficiently computed.

4.3.3 SciPy Methods

For unconstrained optimization, algorithms such as BFGS, Newton-CG, and CG are employed by implicitly enforcing the constraint $\gamma \geq 0$. For constrained optimization, only L-BFGS-B algorithm is used as a benchmark.

1. BFGS and L-BFGS-B

The BFGS (Broyden–Fletcher–Goldfarb–Shanno) method is a quasi-Newton algorithm that iteratively approximates the inverse Hessian matrix using gradient information. It is efficient for medium-scale unconstrained optimization and offers superlinear convergence for smooth problems. However, it requires storing a full $n \times n$ matrix, which becomes impractical for high-dimensional problems. The L-BFGS-B (Limited-memory BFGS with Bounds) method addresses this limitation by storing only a few vectors that represent the approximation implicitly, reducing memory usage.

2. Newton-CG

The Newton-CG (also known as Trust-Region Newton-Conjugate Gradient) method is a second-order optimization algorithm designed for large-scale problems. It minimizes the objective function by solving the Newton system using an inner Conjugate Gradient solver rather than explicitly computing the inverse Hessian. The outer loop checks the

gradient norm, and the inner loop solves for a descent direction up to a residual tolerance.

3. Conjugate-Gradient (Polak-Ribiere) The Polak–Ribiere variant of the Conjugate Gradient method is a first-order method suitable for smooth, unconstrained optimization problems. This method retains conjugacy across iterations, leading to faster convergence than steepest descent. To mitigate loss of conjugacy in non-quadratic or kinked regions, periodic restarts are applied. A tight gradient norm tolerance and relaxed Wolfe condition parameters are recommended for robust line search.

4.4 Data

The data used in this study are Australian Government Bonds obtained from the Australian Securities Exchange (ASX). These bonds are classified as Treasury Bonds, issued by the Australian Government, and are considered to be free of default risk. Compared to corporate bonds, government bonds are backed by the full faith and credit of the federal government, and thus serve as a benchmark for risk-free interest rates. The dataset includes bonds with various maturities and coupon rates, enabling the construction of a comprehensive term structure of interest rates. By analyzing bonds across different maturities, the shape and dynamics of the yield curve can be captured with higher accuracy. The implementation, as of 23 May 2025, is configured to automatically retrieve the latest available data from the ASX. This ensures that the project's yield curve estimation reflects the most up-to-date market conditions.

5 Results & Discussion

5.1 Gradient and Hessian function

The gradient and Hessian of the loss function using three approaches: analytical derivation, numerical approximation, and automatic differentiation via the `Autograd` library. To evaluate efficiency, each method are compared in terms of accuracy and speed, taking the analytical method as the accuracy benchmark and the numerical method pre-compiled in C as the speed baseline. Among the three, `Autograd` achieved the best overall per-

formance. This is likely due to its use of vectorized operations and internal computation graph, which minimizes redundant calculations.

5.2 Result of Optimization Method

Table 1 below summarizes the outcomes of the optimization algorithms applied to estimate the parameters of the Nelson-Siegel model. For each algorithm, the estimated parameters, the final loss value, and the success status of the optimization convergence are reported. In cases where the algorithm failed, the corresponding failure reasons are also noted. The fact that each algorithm estimated substantially different parameter values suggests the existence of multiple local minima in the loss function. This was further supported by experiments where varying the initial values, especially when they were far from the benchmark-led to different optima. This highlights the sensitivity of the model to initial parameter settings. Notably, all implementations based on conjugate-gradient techniques failed to converge. This is likely a consequence of the loss function's non-convex landscape, which can hinder line-search methods that rely solely on gradient information.

Algorithm	f0	f1	f2	gamma	Loss Value	Itr	S/F	Message
DFP	0.871	-0.829	-0.796	108.526	0.000013	24	S	Change of fun is within tolerence
Fletcher-Reeves	-inf	-inf	inf	inf	0.000000	3	S	Norm of gradient is within tolerence
Gradient Projection	0.053	-0.013	-0.032	1.721	2.090541	10	S	Change of x is within tolerence
Zoutendijk	0.042	-0.032	-0.027	1.720	1.362066	209	S	Change of x is within tolerence
CG	0.057	-0.019	-0.019	4.285	0.000032	159	S	Optimization terminated successfully.
Newton-CG	0.066	-0.028	-0.005	14.234	0.645016	300	F	Warning: Maximum number of iterations has been...
L-BFGS-B	0.071	-0.033	-0.007	17.005	0.000013	32	S	Convergence
BFGS	0.066	-0.029	0.000	17.804	6.521601	1	F	Desired error not necessarily achieved due to ...

Table 1: Performance result of manual algorithm and SciPy solvers

To gain a deeper understanding of how each algorithm performs during the optimization process, the loss path is visualized in two different ways. The Figure 3 illustrates the progression of the loss value over iterations, while the Figure 4 shows the loss trajectory with respect to computation time in seconds. As described earlier, some algorithms converged within just a few seconds, whereas others required over a minute to complete. Despite these differences in runtime, most algorithms—except for Gradient Projection and DFP—achieved similar final loss values.

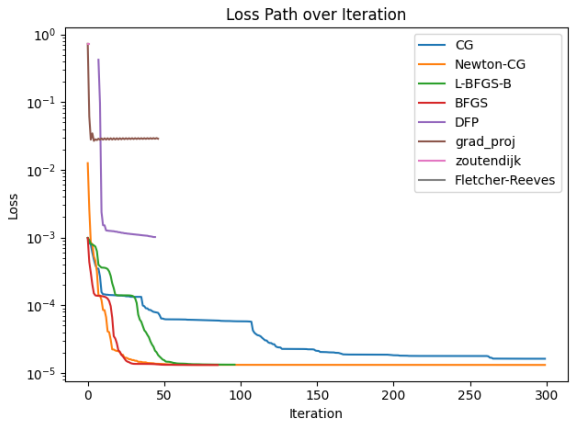

Figure 3: Loss path over iteration of all algorithm

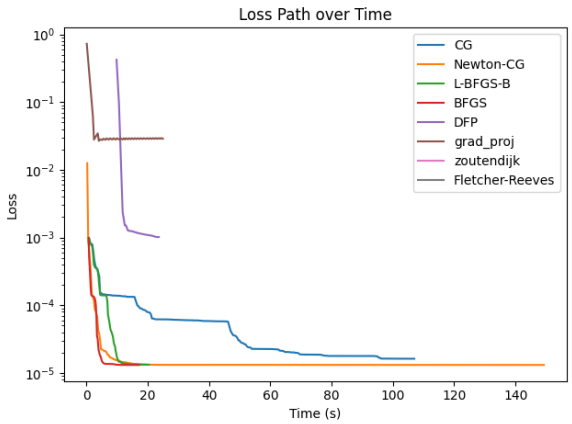

Figure 4: Loss path over time of all algorithm

To identify the best-performing model, the resulting term structures of interest rates are visualised in Figure 5 , which addresses the core objective of the project. In this comparison, algorithms that yielded highly unrealistic γ values are excluded. For instance, although the DFP algorithm was very fast and achieved a low loss value, a transformation to γ to impose a constraint during optimization was applied. However, the resulting γ after optimization was excessively large, leading to an impractical value. In addition, algorithms which showed signs of overfitting are excluded as well, such as those requiring more than 300 iterations for convergence. Among all the evaluated methods, L-BFGS-B delivers the most accurate fit. It not only achieved the lowest loss values but also produced term structures that closely aligned with the benchmark. However, its curves increased more steeply than what was observed in the benchmark, which is likely due to its relatively higher estimated γ value, approximately around 17.

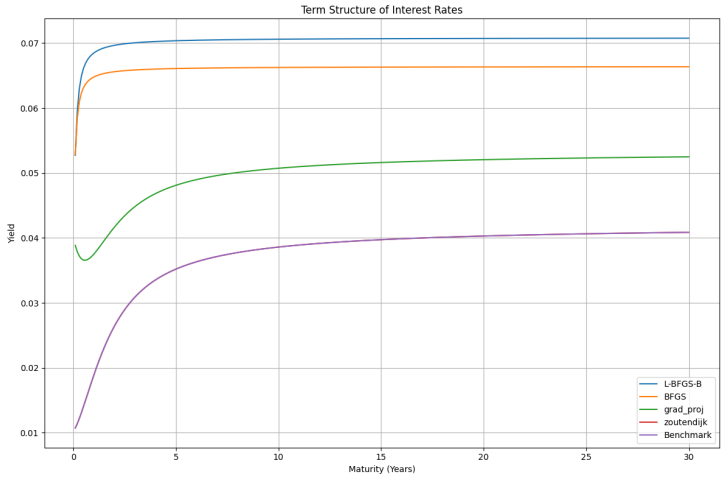

Figure 5: Estimated the term structure of interest rates

6 Conclusion

The objective of this study was to refine the calibration of the Nelson–Siegel yield curve by embedding bid–ask information directly into the loss function. The loss function designed posed challenges due to non-convexity within the range of interest of our model parameters. The loss function is also found to be sensitive to starting point values, and while a fair attempt is carried out to explore a variety of starting points to mitigate this issue, a deeper analysis of initial-value dependence was not conducted. This is because the nature of the yield curve as an economic variable, tends to remain within a relatively stable range. As such, our focus was on identifying parameter estimates that closely match the benchmark curves. Two optimisation strategies were adopted to address the calibration problem. First, by incorporating a constraint on γ directly within the objective function, enabling the use of unconstrained optimisation algorithms. Secondly, boundary conditions were introduced externally to support constrained optimisation. In this context, the the L-BFGS-B algorithm consistently yielded the lowest loss value. These algorithms also produced term structures that closely resemble the benchmark curve, although the long end tended to be steeper due to the large estimated values of γ.

6.1 Limitations and future work

Due to the lack of availability of related research, validation methods on whether the obtained optimum for such loss function designs adequately reflects Australia's term structure of interest rates are limited. This study can also be expanded to explore different loss function designs that balances the tradeoffs between overfitting and retaining bid-ask spread information. Moreover, extending the analysis to bond markets in other countries would test the generality of the proposed calibration framework. A comparative study using multiple sovereign yield curves may also reveal which optimisation techniques and loss structures are most robust to differing liquidity profiles, market conventions, and data quality. Cross-market validation would strengthen confidence in the method's applicability beyond the Australian context.

Appendix A

1. Gradient of the Loss

Define

$$L_j = \Big(\frac{\max\{0, V_j - \text{Ask}_j\}}{A_j}\Big)^2 + \Big(\frac{\max\{0, \text{Bid}_j - V_j\}}{B_j}\Big)^2.$$

Then

$$\frac{\partial L}{\partial \theta} = \sum_{j=1}^{m} \frac{\partial L_j}{\partial \theta}, \qquad \theta \in \{f_0, f_1, f_2, \gamma\}, \tag{1}$$

with

$$\frac{\partial L_j}{\partial \theta} = \begin{cases} \dfrac{V_j - \text{Ask}_j}{A_j^2}\,\dfrac{\partial V_j}{\partial \theta}, & V_j > \text{Ask}_j, \\ 0, & \text{Bid}_j < V_j < \text{Ask}_j, \\ \dfrac{V_j - \text{Bid}_j}{B_j^2}\,\dfrac{\partial V_j}{\partial \theta}, & V_j < \text{Bid}_j. \end{cases}$$

First derivatives of $B(0,T)$.

$$\frac{\partial B}{\partial f_0} = -T\,B(0,T), \tag{2}$$

$$\frac{\partial B}{\partial f_1} = B(0,T)\left(-\gamma + \mathrm{e}^{-T/\gamma}\gamma\right), \tag{3}$$

$$\frac{\partial B}{\partial f_2} = B(0,T)\left(-\gamma + \mathrm{e}^{-T/\gamma}(T+\gamma)\right), \tag{4}$$

$$\frac{\partial B}{\partial \gamma} = B(0,T)\Big[-1 - \mathrm{e}^{-T/\gamma} - \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma}\,f_1 - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2}\big)f_2\Big]. \tag{5}$$

2. Hessian of the Loss

$$\frac{\partial^2 L}{\partial \theta_1 \partial \theta_2} = \begin{cases} \sum_{j=1}^{m} \dfrac{V_j - \text{Ask}_j}{A_j^2}\dfrac{\partial^2 V_j}{\partial \theta_1 \partial \theta_2} + \dfrac{1}{A_j^2}\dfrac{\partial V_j}{\partial \theta_1}\dfrac{\partial V_j}{\partial \theta_2}, & V_j > \text{Ask}_j, \\ 0, & \text{Bid}_j < V_j < \text{Ask}_j, \\ \sum_{j=1}^{m} \dfrac{V_j - \text{Bid}_j}{B_j^2}\dfrac{\partial^2 V_j}{\partial \theta_1 \partial \theta_2} + \dfrac{1}{B_j^2}\dfrac{\partial V_j}{\partial \theta_1}\dfrac{\partial V_j}{\partial \theta_2}, & V_j < \text{Bid}_j, \end{cases} \tag{6}$$

where

$$\frac{\partial^2 V_j}{\partial \theta_1 \partial \theta_2} = 100 \sum_{i=1}^{n_j} c_j \frac{\partial^2 B(0, T_i^{(j)})}{\partial \theta_1 \partial \theta_2} + \frac{\partial^2 B\big(0, T_{n_j}^{(j)}\big)}{\partial \theta_1 \partial \theta_2}.$$

3. Second Derivatives of $B(0,T)$

Let $\theta_1, \theta_2 \in \{f_0, f_1, f_2, \gamma\}$. Every $\partial^2 B / \partial\theta_1 \partial\theta_2$ term is listed below (without algebraic simplification):

$$\frac{\partial^2 B}{\partial f_0^2} = B(0,T)\,T^2, \tag{7}$$

$$\frac{\partial^2 B}{\partial f_0 \partial f_1} = -B(0,T)\,T\big(-\gamma + \mathrm{e}^{-T/\gamma}\gamma\big), \tag{8}$$

$$\frac{\partial^2 B}{\partial f_0 \partial f_2} = -B(0,T)\,T\big(-\gamma + \mathrm{e}^{-T/\gamma}(T+\gamma)\big), \tag{9}$$

$$\frac{\partial^2 B}{\partial f_0 \partial \gamma} = -B(0,T)\Big[T + \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma}\big) f_1 + \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2}\big) f_2\Big], \tag{10}$$

$$\frac{\partial^2 B}{\partial f_1^2} = B(0,T)\big(-\gamma + \mathrm{e}^{-T/\gamma}\gamma\big)^2, \tag{11}$$

$$\frac{\partial^2 B}{\partial f_1 \partial f_2} = B(0,T)\big(-\gamma + \mathrm{e}^{-T/\gamma}\gamma\big)\big(-\gamma + \mathrm{e}^{-T/\gamma}(T+\gamma)\big), \tag{12}$$

$$\frac{\partial^2 B}{\partial f_1 \partial \gamma} = B(0,T)\Big[-1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma} - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma}\big) f_1 - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2}\big) f_2\Big], \tag{13}$$

$$\frac{\partial^2 B}{\partial f_2^2} = B(0,T)\big(-\gamma + \mathrm{e}^{-T/\gamma}(T+\gamma)\big)^2, \tag{14}$$

$$\frac{\partial^2 B}{\partial f_2 \partial \gamma} = B(0,T)\Big[-1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2} + \big(-\gamma + \mathrm{e}^{-T/\gamma}(T+\gamma)\big) \tag{15}$$

$$- \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma}\big) f_1 - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2}\big) f_2\Big], \tag{16}$$

$$\frac{\partial^2 B}{\partial \gamma^2} = B(0,T)\Big[\mathrm{e}^{-T/\gamma}\frac{T^2}{\gamma^3} f_1 - 2\,\mathrm{e}^{-T/\gamma}\frac{T}{\gamma^2} - \mathrm{e}^{-T/\gamma}\frac{T^2(T+\gamma)}{\gamma^4} \tag{17}$$

$$+ 2\,\mathrm{e}^{-T/\gamma}\frac{T(T+\gamma)}{\gamma^3} f_2 - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T}{\gamma}\big) f_1 - \big(1 + \mathrm{e}^{-T/\gamma} + \mathrm{e}^{-T/\gamma}\tfrac{T(T+\gamma)}{\gamma^2}\big) f_2\Big]. \tag{18}$$

Appendix B: Code Repository

The source code used for all experiments and visualizations in this study is available at: `https://github.com/Thaigithub/UTS_PO_Assignment`

References

Nelson, C. R. and Siegel, A. F. (1987). Parsimonious modeling of yield curves. *Journal of Business*, 60(4), 473–489.

Gomes-Gonçalves, E., Gzyl, H. and Mayoral, S. (2017). Calibration of short-rate term-structure models from bid–ask coupon-bond prices. *Physica A: Statistical Mechanics and its Applications*, 492, 1456–1472.

Lapshin, V. and Sohatskaya, S. (2020). Choosing the weighting coefficients for estimating the term structure from sovereign bonds. *International Review of Economics & Finance*, 70, 635–648.

Diebold, F. X. and Li, C. (2006). Forecasting the term structure of government bond yields. *Journal of Econometrics*, 130(2), 337–364.