Computing Quasi-Normal Modes of Schwarzschild Black Holes with Physics-Informed Neural Networks

Supervisor Progress Presentation

Jonathan Chung

University of Cambridge

25 February 2026

Outline

What Are Quasi-Normal Modes?

- A perturbed Schwarzschild black hole emits gravitational radiation as it settles down — the **ringdown**.
- The ringdown consists of **quasi-normal modes** (QNMs): damped sinusoids

$$\Phi(t) \propto e^{-t/\tau} \cos(\omega t)$$

- The frequency ω and decay time τ depend **only on the black hole mass** M (for Schwarzschild).
- **Goal:** Solve the perturbation equations numerically to extract ω and τ.

From Einstein's Equations to a Master Equation

Start with the Schwarzschild metric and add a small perturbation:

$$g_{\mu\nu} = g^0_{\mu\nu}(\text{Schwarzschild}) + h_{\mu\nu}(\text{small})$$

Key simplifications:

1. Linearise Einstein's equations in $h_{\mu\nu}$.
2. Decompose $h_{\mu\nu}$ in **tensor spherical harmonics** — angular dependence separates.
3. Construct a gauge-invariant **master function** Φ from the metric perturbation components.

Result: a single 1+1D wave equation for each angular mode ℓ:

$$\boxed{-\frac{\partial^2\Phi}{\partial t^2} + \frac{\partial^2\Phi}{\partial x^2} - V(r)\,\Phi = 0}$$

where $x = r + 2M\ln\left(\frac{r}{2M} - 1\right)$ is the **tortoise coordinate**.

The Potentials

The potential $V(r)$ depends on the **parity** of the perturbation:

Odd-parity (axial) — Regge–Wheeler:

$$V_{\mathrm{RW}} = \left(1 - \frac{2M}{r}\right)\left[\frac{\ell(\ell+1)}{r^2} - \frac{6M}{r^3}\right]$$

Even-parity (polar) — Zerilli:

$$V_{\mathrm{Z}} = \left(1 - \frac{2M}{r}\right)\frac{2n^2(n+1)r^3 + 6n^2Mr^2 + 18nM^2r + 18M^3}{r^3(nr+3M)^2}$$

where $2n = (\ell-1)(\ell+2)$.

- Both potentials peak near $x \sim 1M$ and vanish at the horizon and infinity.
- A transformation connects them (Chandrasekhar) $\Rightarrow$ **same QNMs**.
- We focus on the **Zerilli potential** with $\ell = 2$ (dominant mode).

Boundary & Initial Conditions

Boundary conditions (Sommerfeld — radiation conditions):

$$\begin{aligned} \text{Horizon } (x \to -\infty): &\quad (\partial_t - \partial_x)\,\Phi = 0 \quad \text{(ingoing)} \\ \text{Infinity } (x \to +\infty): &\quad (\partial_t + \partial_x)\,\Phi = 0 \quad \text{(outgoing)} \end{aligned}$$

Initial conditions — outgoing Gaussian pulse:

$$\Phi(x,0) = \exp\left[-\frac{(x-4M)^2}{(5M)^2}\right]$$

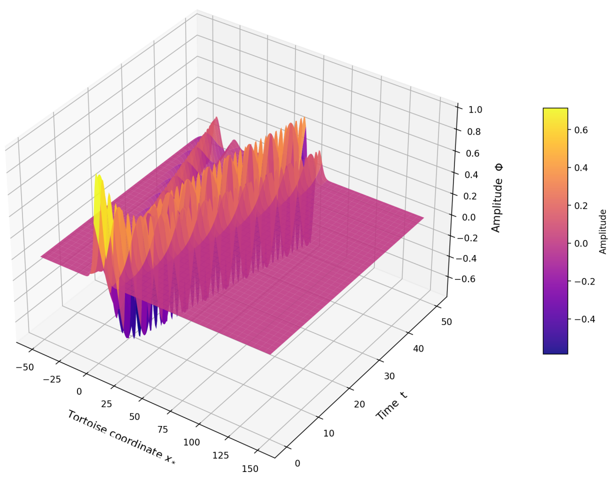
$$\partial_t \Phi(x,0) = -\partial_x \Phi(x,0) \quad \text{(purely right-moving)}$$

Domain: $x/M \in [-50, 150]$, $\quad t/M \in [0, 50]$.

The pulse scatters off the potential barrier $\Rightarrow$ ringdown at late times.

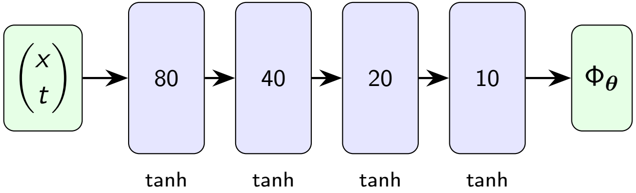
Ringdown Waveform — 3D Visualisation

3D Waveform of Schwarzschild Black Hole Ringdown (FD Reference)



Physics-Informed Neural Networks — Core Idea

Instead of a grid, train a neural network $\Phi_{\boldsymbol{\theta}}(x, t)$ to satisfy the PDE, BCs, and ICs simultaneously.



$\begin{pmatrix} x \\ t \end{pmatrix}$
80
40
20
10
$\Phi_{\boldsymbol{\theta}}$
tanh
tanh
tanh
tanh

- 4 hidden layers, tanh activation, Glorot uniform initialisation.
- Output transform: $\Phi_{\boldsymbol{\theta}} = A \tanh(\text{raw output})$ bounds the solution to $[-1, 1]$.
- Derivatives via **automatic differentiation** (exact, no discretisation error).
- **4,521 trainable parameters.**

The Loss Function

Train by minimising a weighted sum of 7 terms:

$$\mathcal{L} = \boldsymbol{\lambda} \cdot \Big[\underbrace{\mathcal{L}_r,\ \mathcal{L}_{r_x},\ \mathcal{L}_{r_t}}_{\text{PDE residual}},\ \underbrace{\mathcal{L}_{ic},\ \mathcal{L}_{iv}}_{\text{initial conds.}},\ \underbrace{\mathcal{L}_{bl},\ \mathcal{L}_{br}}_{\text{boundary conds.}} \Big]$$

Term	**Penalises**	**Points**	$\boldsymbol{\lambda}$
$\mathcal{L}_r$	$\Phi_{tt} - \Phi_{xx} + V\Phi \neq 0$	$N_r = 32{,}000$	100
$\mathcal{L}_{r_x}, \mathcal{L}_{r_t}$	Gradients of residual (gPINN)	same	100, 100
$\mathcal{L}_{ic}$	Initial profile mismatch	$N_i = 800$	1
$\mathcal{L}_{iv}$	Initial velocity mismatch	same	100
$\mathcal{L}_{bl}, \mathcal{L}_{br}$	Sommerfeld BC violations	$N_b = 400$	1, 1

- **Phase problem:** $\Phi(x, t + \alpha)$ is also a solution $\Rightarrow$ weight λ_{iv} heavily to lock the correct phase.
- Total: **33,600 collocation points** (vs. 500,000 grid points for FD).

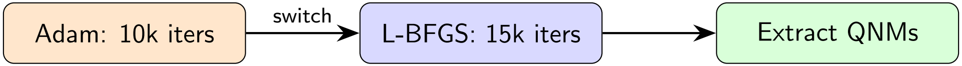
Two-Phase Training

Phase 1: Adam (10,000 iters)

- Stochastic gradient descent with adaptive learning rate (lr $= 10^{-3}$).
- Good at rough, global exploration.
- Uniform point resampling every 100 iters.

Phase 2: L-BFGS (15,000 iters)

- Quasi-Newton: approximates inverse Hessian from gradient history.
- Fast, precise convergence near minima.
- Sensitive to point changes $\Rightarrow$ only uniform resampling here.



Finite Difference Baseline

For comparison, we solve the same PDE with standard numerical methods:

- Uniform mesh: $N_x = 1{,}000$ points, $\Delta x = 0.2M$.
- Method of lines + 4th-order Runge–Kutta, $\Delta t = 0.1M$.
- 500 time steps $\Rightarrow N_F = 500{,}000$ grid points.

The FD solution serves as **ground truth** for evaluating the PINN.

Key comparison:

	FD	**PINN**
Representation	Grid values $\Phi_{i,j}$	Neural network $\Phi_\theta(x,t)$
Derivatives	FD stencils (approx.)	Autograd (exact)
Points	500,000	33,600
Time-stepping	Explicit RK4	None (global solve)

Extracting Quasi-Normal Modes

Sample $\Phi(x_q, t)$ at observation point $x_q = 10M$, restrict to late times.

Method 1 — FFT + Envelope:

- ω: FFT of $\Phi(x_q, t) \;\Rightarrow\;$ peak frequency.
- τ: Linear fit to $\ln|\text{envelope peaks}|$ vs. $t \;\Rightarrow\;$ slope $= -1/\tau$.

Method 2 — Direct curve fit:

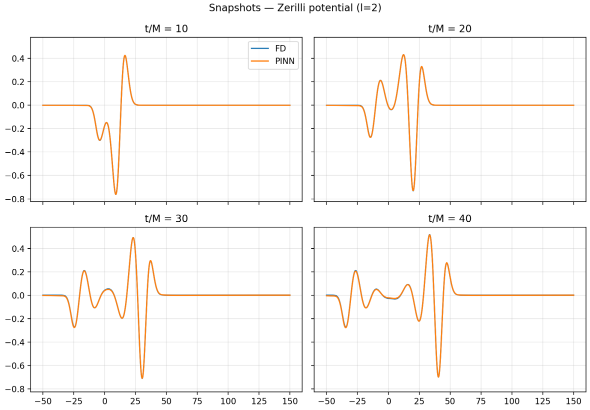
- Fit $\Phi \approx A\, e^{-t/\tau} \cos(\omega t + \phi)$ directly via nonlinear least squares.

Compare extracted (ω, τ) against theoretical values (Leaver 1985):

ℓ	ωM	τ/M
2	0.3737	11.241

Paper Reproduction: Baseline Results

Reproduced the setup of Patel, Aykutalp & Laguna (2024) with our corrected outgoing initial velocity profile.



Waveform snapshots: PINN vs. FD (uniform sampling)

Zerilli $\ell = 2$ QNM Extraction

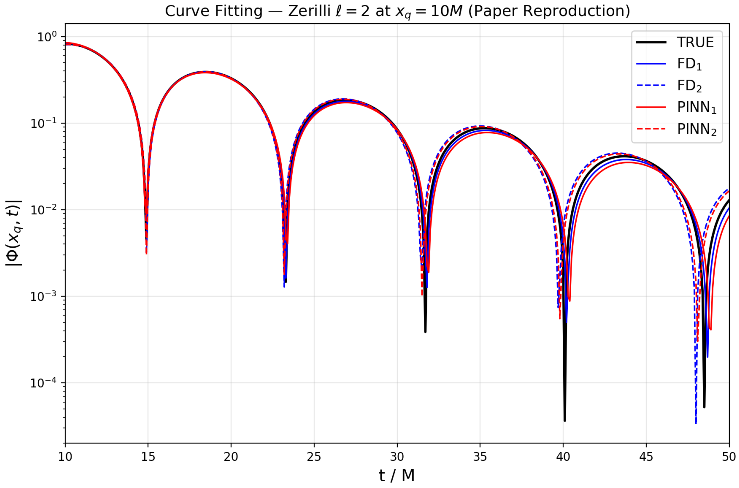
	τ/M	ωM
TRUE	11.241	0.3737
FD$_1$	10.929 (2.78)	0.3716 (0.56)
FD$_2$	11.465 (2.00)	0.3798 (1.63)
PINN$_1$	10.671 (5.07)	0.3697 (1.08)
PINN$_2$	11.410 (1.50)	0.3784 (1.25)

Parentheses: % error vs. Leaver 1985.

Subscript 1 = FFT + envelope.

Subscript 2 = direct curve fit.

Curve Fitting — Paper Reproduction



$\log|\Phi|$ vs. t/M at $x_q = 10M$. All five reconstructed damped cosines $A\,e^{-t/\tau}\cos(\omega t + \phi)$ overlap closely — both FD and PINN extract QNMs within $\sim$1–4% of the Leaver (1985) values.

QNM Accuracy: Our Results vs. the Paper

Zerilli $\ell = 2$ — τ/M and ωM percentage errors

	Paper (Patel et al.)		**Ours**	
	τ/M (% err)	ωM (% err)	τ/M (% err)	ωM (% err)
FD$_1$	10.804 (3.89)	0.370 (0.90)	10.929 (2.78)	0.372 (0.56)
FD$_2$	11.073 (1.49)	0.378 (1.26)	11.465 (2.00)	0.380 (1.63)
PINN$_1$	10.089 (10.25)	0.375 (0.29)	10.671 (**5.07**)	0.370 (1.08)
PINN$_2$	9.620 (14.42)	0.376 (0.52)	11.410 (**1.50**)	0.378 (1.25)

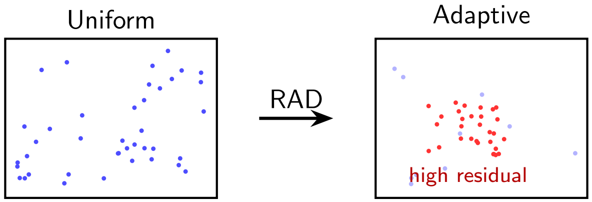
- **Major improvement:** PINN$_2$ decay time error drops from 14.42% $\rightarrow$ 1.50% (**9.6**$\times$ more accurate).
- PINN$_1$ τ error also improves: 10.25% $\rightarrow$ 5.07% (2.0$\times$).
- Likely cause: our corrected outgoing initial velocity $\partial_t \Phi = -\partial_x \Phi$ (Sommerfeld condition).

Our Improvement: Residual-Adaptive Distribution (RAD)

Problem: Uniform sampling wastes points in empty regions where $\Phi \approx 0$.

RAD (Wu et al. 2023): Every P training steps during Adam—

1. Generate 100,000 random candidate points.
2. Evaluate PDE residual $|r_i|$ at each candidate.
3. Compute sampling probability: $p_i \propto |r_i|^k + c$.
4. Sample 32,000 points from candidates $\Rightarrow$ replace all domain points.

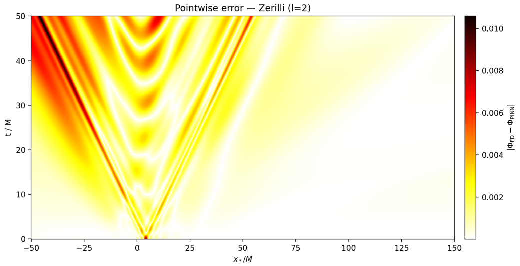


RAD Tuning Experiments

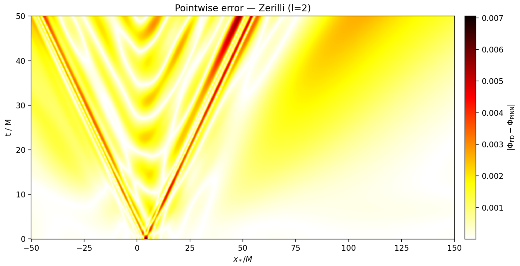
Configuration	k	c	**Final Loss**	**RMSD**	**RL2**
Uniform (paper repro)	—	—	1.18×10^{-6}	0.00183	1.30%
RAD baseline	1	1	7.40×10^{-7}	0.00115	0.82%
RAD aggressive	2	0.1	9.07×10^{-7}	**0.00095**	**0.67%**
RAD $k=2$, $P=500$	2	0.1	1.06×10^{-6}	0.00105	0.75%
RAD + anchor (20%)	1	1	8.20×10^{-7}	0.00141	1.00%

- RAD reduces RMSD by **37%** over uniform sampling.
- Aggressive RAD ($k=2$) achieves the **best RMSD** (0.00095).
- Anchor retention has the **second-lowest loss** (8.20×10^{-7}) but the **highest RMSD** among RAD runs — suggests over-fitting to anchor points at the expense of global accuracy.

Spatial Error Distribution



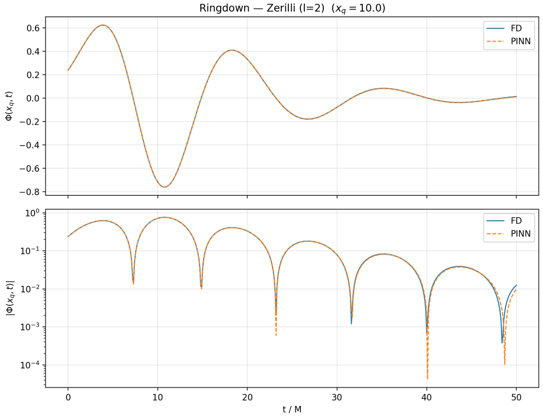
Uniform sampling



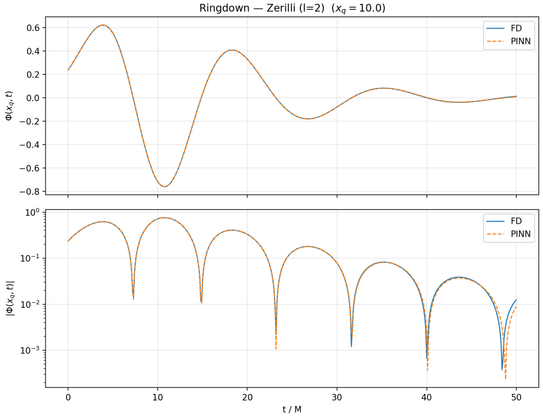
RAD adaptive sampling

RAD reduces the error concentration at the **wavefront leading edge**.

Ringdown Waveform at $x_q = 10M$



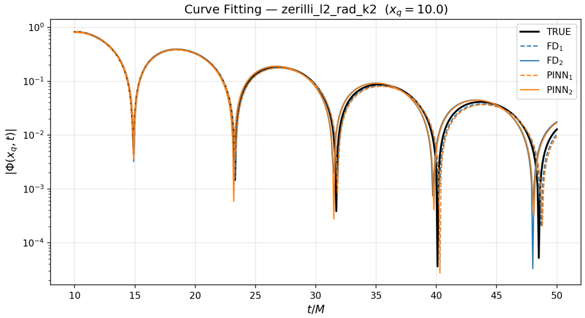
RAD baseline: PINN vs. FD (raw waveforms)



RAD $k = 2$: PINN vs. FD (raw waveforms)

Top: linear scale; Bottom: $\log|\Phi|$. Both variants closely match the FD reference.

Curve Fitting — RAD $k = 2$



log $|\Phi|$ vs. t/M at $x_q = 10M$ (RAD $k = 2$)

Zerilli $\ell = 2$ — QNM Extraction

	τ/M	ωM
TRUE	11.241	0.3737
FD$_1$	10.929 (2.78)	0.3716 (0.56)
FD$_2$	11.465 (2.00)	0.3798 (1.63)
PINN$_1$	10.831 (3.65)	0.3705 (0.85)
PINN$_2$	11.443 (1.80)	0.3790 (1.43)

FD$_{1,2}$ identical to paper repro (same FD solver).

PINN$_{1,2}$ from RAD $k = 2$ run.

- RAD PINN$_2$: τ error 1.80% vs. 1.50% (paper repro) — comparable accuracy.
- RAD primarily improves **spatial** accuracy (RMSD), not QNM extraction.

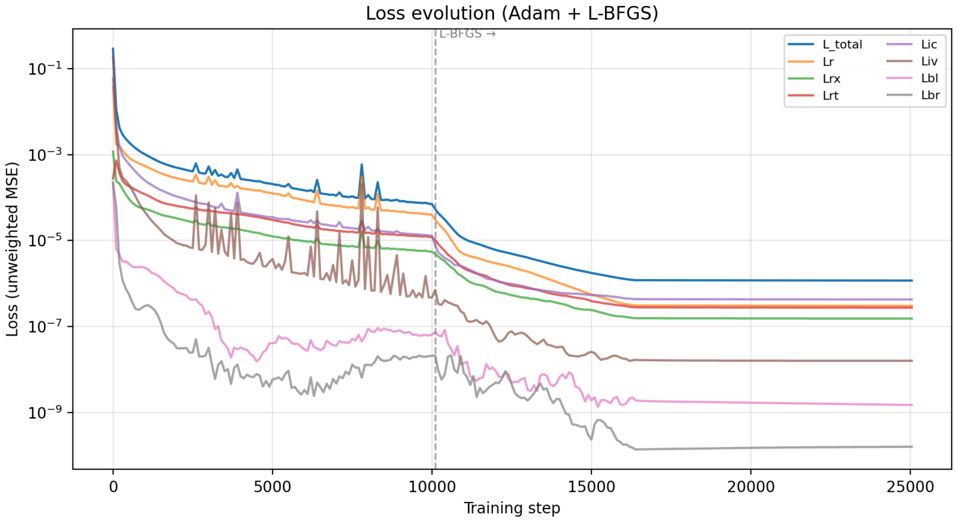
Summary of Findings

1. **PINNs can compute QNMs** of Schwarzschild black holes to within $\sim$1–2% of known values, using $15\times$ fewer collocation points than FD.
2. **RAD adaptive sampling** improves solution accuracy by $\sim$37% (RMSD) over uniform sampling, by concentrating points where the PDE residual is largest.
3. **Aggressive RAD** ($k = 2$) provides the best spatial accuracy (lowest RMSD/RL2). Anchor retention has the second-lowest loss but the highest RMSD among RAD runs — over-fitting to anchor points.
4. **Exponential reweighting fails** — naively amplifying late-time residuals destabilises training (τ error $>$20%).

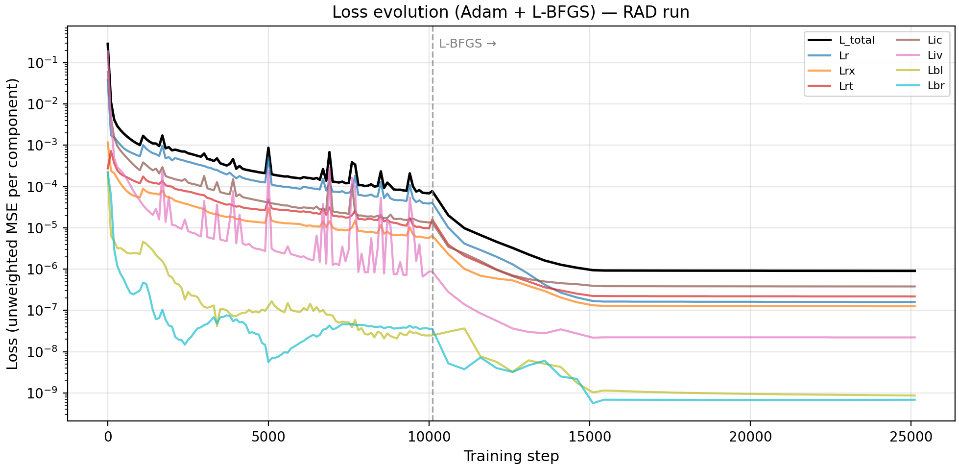
Thank you

Questions?

Appendix: Loss History — Reproduction Run (Uniform Sampling)



Appendix: Loss History — RAD Adaptive Sampling



Appendix: QNM Extraction — Method 1 (FFT + Envelope)

Extract ω and τ **independently** from the late-time waveform $\Phi(x_q, t)$.

Step 1 — Estimate ω via FFT:

1. Window the signal between t_{start} and t_{end}; subtract the mean.
2. Apply a Hann window to reduce spectral leakage.
3. Zero-pad by $64\times$ and compute the real FFT.
4. Locate the magnitude peak; refine with **parabolic interpolation** for sub-bin accuracy.
5. Convert: $\omega = 2\pi f_{\text{peak}}$.

Step 2 — Estimate τ via log-linear envelope fit:

1. Find all local maxima of $|\Phi(x_q, t)|$ (envelope peaks).
2. Since the envelope decays as $A\,e^{-t/\tau}$, take the logarithm:

$$\ln|y_{\text{peaks}}| = \ln A - \frac{t}{\tau}.$$

3. A linear fit gives slope $m = -1/\tau$, hence $\tau = -1/m$.

Appendix: QNM Extraction — Method 2 (Nonlinear Curve Fit)

Fit the full damped-cosine model **simultaneously** to the waveform:

$$\Phi(x_q, t) \approx A\, e^{-t/\tau} \cos(\omega\, t + \phi).$$

Procedure:

1. Obtain initial guesses from Method 1: ω_0, τ_0.
2. Set $A_0 = \max|\Phi|$, $\phi_0 = 0$.
3. Run `scipy.optimize.curve_fit` (Levenberg–Marquardt) with four free parameters (A, τ, ω, ϕ).

Advantages over Method 1:

- All four parameters are fitted **jointly**, so correlations between ω and τ are captured.
- Uses **every data point**, not just envelope peaks $\Rightarrow$ typically more accurate.

Disadvantage:

- Requires good initial guesses; can fail to converge if the signal is noisy or the window is too short.

Appendix: L-BFGS — Motivation & Core Idea

Why two optimisers?

- **Adam** (Phase 1): stochastic, handles noisy/flat loss landscapes well; reaches a rough neighbourhood of the minimum quickly.
- **L-BFGS** (Phase 2): quasi-Newton method that uses curvature information to converge much faster once we are near a minimum.

Full BFGS recap. Newton's method updates parameters via $\theta_{k+1} = \theta_k - H_k^{-1} \nabla \mathcal{L}_k$, where H_k is the Hessian. Computing and inverting H is $O(n^2)$ storage and $O(n^3)$ per step (n = number of parameters) — **impractical for neural networks**.

BFGS (Broyden–Fletcher–Goldfarb–Shanno) avoids forming the Hessian explicitly. Instead it maintains an *approximate inverse Hessian* B_k^{-1} and updates it each step using only gradient differences:

$$s_k = \theta_{k+1} - \theta_k, \qquad y_k = \nabla \mathcal{L}_{k+1} - \nabla \mathcal{L}_k.$$

Appendix: L-BFGS — The BFGS Update

The rank-2 update of the *approximate inverse Hessian* B_k^{-1}:

$$B_{k+1}^{-1} = \left(I - \rho_k\, s_k\, y_k^T\right) B_k^{-1} \left(I - \rho_k\, y_k\, s_k^T\right) + \rho_k\, s_k\, s_k^T, \quad \rho_k = \frac{1}{y_k^T s_k}.$$

This ensures B_{k+1} satisfies the **secant condition** $B_{k+1}\, s_k = y_k$ while staying symmetric positive-definite.

The search direction is then:

$$d_k = -B_k^{-1}\, \nabla \mathcal{L}_k$$

followed by a **line search** along d_k to find the step size.

Problem: Still $O(n^2)$ storage for the dense matrix B^{-1} — too expensive when n is the number of neural network weights.

Appendix: L-BFGS — Limited-Memory Variant

L-BFGS (Limited-memory BFGS) solves the storage problem:

- Store only the last m pairs $\{s_k, y_k\}$ (typically $m = 20$–50).
- Reconstruct the matrix–vector product $B_k^{-1}\nabla\mathcal{L}$ on-the-fly via a **two-loop recursion** — never form B^{-1}.
- Storage: $O(mn)$ instead of $O(n^2)$; each step is $O(mn)$.

Two-loop recursion (Nocedal 1980):

1. *Backward pass*: for $i = k-1, \dots, k-m$, compute scalar $\alpha_i = \rho_i\, s_i^T q$ and set $q \leftarrow q - \alpha_i\, y_i$.
2. Scale: $r \leftarrow H_k^0\, q$ (diagonal initialisation).
3. *Forward pass*: for $i = k-m, \dots, k-1$, set $r \leftarrow r + s_i(\alpha_i - \rho_i\, y_i^T r)$.
4. Search direction: $d_k = -r$.

In our pipeline:

- Adam runs for 10 k steps $\Rightarrow$ L-BFGS refines for 15 k steps.
- L-BFGS uses *full-batch* gradients (all 33 600 points), giving accurate curvature estimates.
- Typically reduces the loss by a further 1–2 orders of magnitude.

Appendix: Finite Difference Solver — Method of Lines

Rewrite the PDE $\Phi_{tt} - \Phi_{xx} + V\Phi = 0$ as a **first-order system**:

$$u = \Phi, \qquad v = \Phi_t, \qquad \begin{cases} u_t = v, \\ v_t = u_{xx} - V(x)\, u. \end{cases}$$

Spatial discretisation (2nd-order central differences):

$$u_{xx}\Big|_i \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}, \qquad 1 \le i \le N_x - 2.$$

At boundaries ($i = 0$ and $i = N_x - 1$): 2nd-order **one-sided** stencils, e.g.

$$u_{xx}\Big|_0 \approx \frac{2u_0 - 5u_1 + 4u_2 - u_3}{\Delta x^2}.$$

This converts the PDE into a system of ODEs in time: $\dot{\mathbf{U}}(t) = \mathbf{F}\big(\mathbf{U}(t)\big)$, where $\mathbf{U} = (u_0, \dots, u_{N_x - 1},\ v_0, \dots, v_{N_x-1})$.

Appendix: Finite Difference Solver — RK4 & Boundary Conditions

Time integration — Classical RK4:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \tfrac{\Delta t}{6}\big(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\big),$$

where $\mathbf{k}_1 = \mathbf{F}(\mathbf{U}^n)$, $\mathbf{k}_2 = \mathbf{F}(\mathbf{U}^n + \tfrac{\Delta t}{2}\mathbf{k}_1)$, etc.
4th-order accuracy in time; requires $\Delta t/\Delta x \leq 1$ (CFL condition). We use $\Delta x = 0.2M$, $\Delta t = 0.1M$ $\Rightarrow$ CFL ratio $= 0.5$.

Sommerfeld radiative BCs (applied at every RK4 sub-step):

$$\begin{aligned}\text{Left } (x \to -\infty):\ (\partial_t - \partial_x)\Phi = 0 \quad &\Rightarrow \quad v_0 = u_x\big|_0,\\ \text{Right } (x \to +\infty):\ (\partial_t + \partial_x)\Phi = 0 \quad &\Rightarrow \quad v_{N_x-1} = -u_x\big|_{N_x-1}.\end{aligned}$$

Spatial derivatives u_x at boundaries use 2nd-order one-sided stencils:

$$u_x\big|_0 \approx \frac{-3u_0 + 4u_1 - u_2}{2\Delta x}, \qquad u_x\big|_{N_x-1} \approx \frac{3u_{N_x-1} - 4u_{N_x-2} + u_{N_x-3}}{2\Delta x}.$$