STAT600 Final Project: R Package Proposal

Name: MISHRA, SHLOK **GitHub Account:** https://github.com/shlokmishra
Tentative package name and title:
autodiffr — A tidy, torch-powered toolkit for Auto-Diff M-Estimation and MLE in R

Short description of context

When models require custom likelihoods or M-estimators (e.g., zero-inflation, censoring, mixtures), typical R workflows rely on numeric derivatives (`optim+numDeriv`) or C++ templates (TMB).[1] autodiffr offers a high-level, tidy API for frequentist estimation using automatic differentiation via `torch`,[2] providing stable gradients/Hessians, smooth constraint handling (positivity, simplex, SPD), and classical inference (information matrices, sandwich SEs,[3] LR/score tests). *Example use:* fit a zero-inflated Poisson with positivity constraints, then report profile likelihood CIs—without writing C++.

Short description of intended functionality:

Users provide `loglik(`θ`, data)` or estimating equations `psi(`θ`, data)` plus starting values; autodiffr returns estimates, variance, diagnostics, and profile-based intervals. Constraints are enforced via smooth reparameterizations (log/softplus, softmax, Cholesky) with automatic Jacobian corrections. Outputs are tidy tibbles and S3 objects with `tidy/glance/augment/autoplot`.

Planned functions (names + purpose):

- `optim_mle()` — Maximize user log-likelihood with torch auto-diff (LBFGS/Adam), honoring constraints; return fit + convergence summary.
- `optim_mest()` — Fit M-estimators from `psi()`; compute Godambe (sandwich) variance with options for small-sample corrections.
- `vcov_info()` — Observed/expected information or sandwich variance; return `vcov`, SEs, condition diagnostics.
- `profile_lik()` — Profile a parameter; return grid, LR stats, CI cutoffs; `autoplot()` for profile curves.
- `constraints()` — Helpers: `positive()`, `simplex()`, `corr_matrix()` (Cholesky) with Jacobian adjustments.
- `check_grad()` — Compare auto-diff vs finite differences; report max relative error and flag issues.

Required inputs:

- Data: `data.frame`/tibble or list; optional formula interface.
- Start values: named `numeric()`; constraint specs via helper constructors.
- Functions: `loglik(theta, data)` (scalar) or `psi(theta, data)` (moment vector).
- Options: optimizer (`"lbfgs"`—`"adam"`), tolerances, max iterations, optional minibatching.

Expected outputs:

- S3 fit object: `coef`, `vcov`, convergence info, gradient/Hessian checks; `broom` methods (`tidy/glance/augment`).
- Tidy tibbles for profiles/diagnostics; `ggplot` objects via `autoplot()` (e.g., profile curves, convergence plots).
- Informative messages/warnings (non-convergence, ill-conditioning, gradient mismatch); optional CSVs for profile grids via `path=`.

[1] Kristensen et al. (2016), *J. Stat. Softw.*

[2] Paszke et al. (2019), NeurIPS

[3] White (1982), *Econometrica*; Huber (1964), *Ann. Math. Stat.*