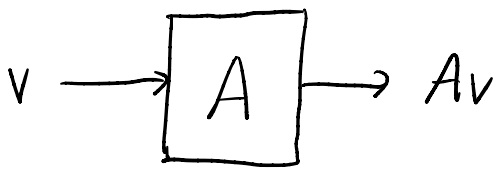
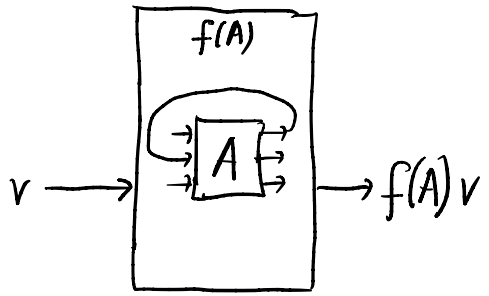
Large-scale linear algebra with curvature matrices

Overview: linear operators

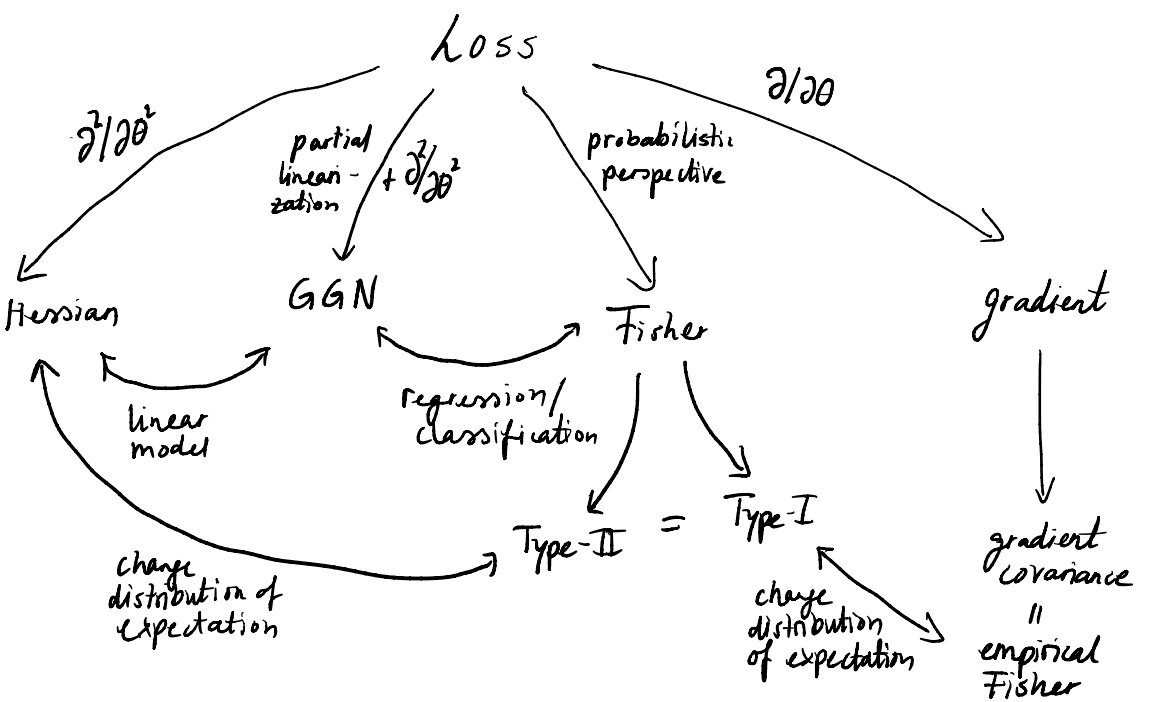


$$A(v_1 + v_2) = Av_1 + Av_2$$
$$A(cv) = cAv$$



f: $\mathrm{Tr}(\cdot)$, $\mathrm{eig}(\cdot)$, $\mathrm{diag}(\cdot)$, $\mathrm{linsolve}(\cdot)$, $(\cdot)^{-1}$

Overview: DL curvature matrices and their relation

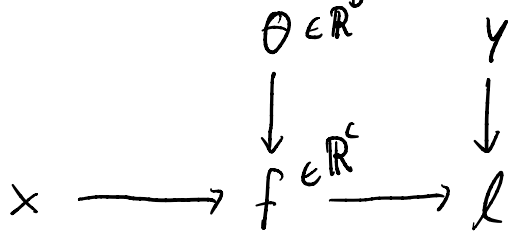


DL

Curvature

matrices

Loss, gradient, Hessian



$+ \quad \mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$

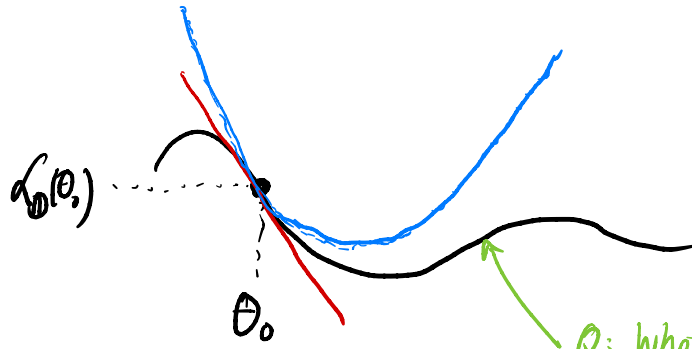
$$\mathcal{L}_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \ell(f(x_n, \theta), y_n) \quad \in \mathbb{R}$$

$$g_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \ell(f(x_n, \theta), y_n) \quad \in \mathbb{R}^D$$

$$H_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta^2 \ell(f(x_n, \theta), y_n) \quad \in \mathbb{R}^{D \times D}$$

Motivation: Taylor expansion

$$\mathcal{L}_{\mathcal{D}}(\theta - \theta_0) = \mathcal{L}_{\mathcal{D}}(\theta_0) + g_{\mathcal{D}}(\theta_0)^T (\theta - \theta_0) + \frac{1}{2} (\theta - \theta_0)^T H_{\mathcal{D}}(\theta_0) (\theta - \theta_0) + \mathcal{O}\left((\theta - \theta_0)^3\right)$$



Use cases:

- Optimization (fit local quadratic, jump to minimum)
- Pruning (find weight that least affects loss when removed)
- Sharpness & influence functions (sensitivity of loss w.r.t. perturbations in the data/parameters)
- Laplace approximations (fit a Gaussian to a more complicated distribution)

Pros and cons of the Hessian:

- Cannot touch the full matrix explicitly ($D \times D$)
- Can multiply with the Hessian efficiently using first-order autodiff! (double-backward)

$$\left(\frac{\partial^2}{\partial \theta^2}\mathcal{L}\right) v = \frac{\partial}{\partial \theta}\left[\left(\frac{\partial}{\partial \theta}\mathcal{L}\right) \cdot v\right] = \frac{\partial}{\partial \theta}\left[g \cdot v\right]$$

- Hessian can be indefinite because $\ell \circ f$ is usually non-convex
  (ℓ: convex, f: non-convex)

Generalized Gauss-Newton (GGN) matrix

- Compositions of convex functions are convex
  → can we convexify f? Yes, through linearization around some anchor θ_0:

$$f(\theta) \rightarrow f_{\theta_0}^{\text{lin}}(\theta) = f(\theta_0) + \underbrace{J_{\theta_0} f(\theta_0)}_{C \times D} (\theta - \theta_0), \quad [\cdot]_{ij} = \frac{\partial f_i}{\partial \theta_j}$$

$$H_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta^2 (\ell_n \circ f_n)(\theta)$$

Hessian of the partially linearized loss

$$G_{\mathcal{D}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \left[\nabla_\theta^2 (\ell_n \circ f_{\theta_0, n}^{\text{lin}})(\theta) \right]_{\theta_0 = \theta}$$

"GGN"

$$= \frac{1}{N} \sum_{n=1}^{N} \underbrace{(J_\theta f_n)^T}_{D \times C} \underbrace{(\nabla^2_{f_n} \ell_n)}_{C \times C} \underbrace{(J_\theta f_n)}_{C \times D}$$

- Positive semi-definite
- GGN-vector products trough JVP, HVP, VJP
- = Hessian if f is linear in θ (e.g. linear regression)

// Prelude to Fisher: Risk minimization

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \ell(f(x_n, \theta), y_n)$$ (massage into an expectation)

Assume $\exists$ data-generating process $p_{data}(x, y)$ and we want to minimize the expected risk

$$\mathcal{L}_{p_{data}}(\theta) = \mathbb{E}_{p_{data}(x,y)}\left[\ell(f(x, \theta), y) \right] \quad \text{w.r.t. } \theta.$$

↳ p_{data} is intractable though. That's why we collected a data set, which approximates

$$p_{data}(x, y) \approx p_{\mathcal{D}}(x, y) := \frac{1}{N} \sum_{n=1}^{N} \delta(x - x_n) \delta(y - y_n)$$

↑

"empirical data distribution", tractable

We then have

$$\mathcal{L}_{\mathcal{D}}(\theta) = \mathcal{L}_{p_{\mathcal{D}}}(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x,y)}\left[\ell(f(x,\theta),y)\right]$$

for the empirical risk.

Connection to maximum likelihood

Want to learn a parametrized density $p_\theta(x,y)$ that approximates $p_{data}(x,y)$:

$$\underset{\theta}{\text{minimize}} \; KL\left(p_{data}(x,y) \,\|\, p_\theta(x,y)\right)$$

$$\Leftrightarrow \underset{\theta}{\text{minimize}} \; \mathbb{E}_{p_{data}(x,y)}\left[-\log p_\theta(x,y)\right]$$

(we will only model $y|x$ with parameters, so $p_\theta(x,y) = p_\theta(y|x)\, p_{data}(x)$, we don't have access to p_{data}, so $p_{data} \hookrightarrow p_{\mathcal{D}}$)

$$\underset{\theta}{\text{minimize}} \; \frac{1}{N} \sum_{n=1}^{N} \underbrace{-\log p_\theta(y_n \mid x_n)}$$

This looks like $\ell(f(x_n,\theta), y_n)$ from above, where ℓ is a negative log-likelihood

Rewrite $p_\theta(y \mid x) = r(y \mid f(x,\theta))$, then we can show

1) $r(y \mid f(x,\theta)) = \mathcal{N}(y \mid \mu = f(x,\theta), \Sigma = I)$

$\Leftrightarrow \; -\log r = \ell(\text{MSELoss})$

2) $r(y \mid f(x,\theta)) = \text{Cat}(y \mid \pi = \text{softmax}(f(x,\theta)))$

$y \in \{1, \dots, C\}$

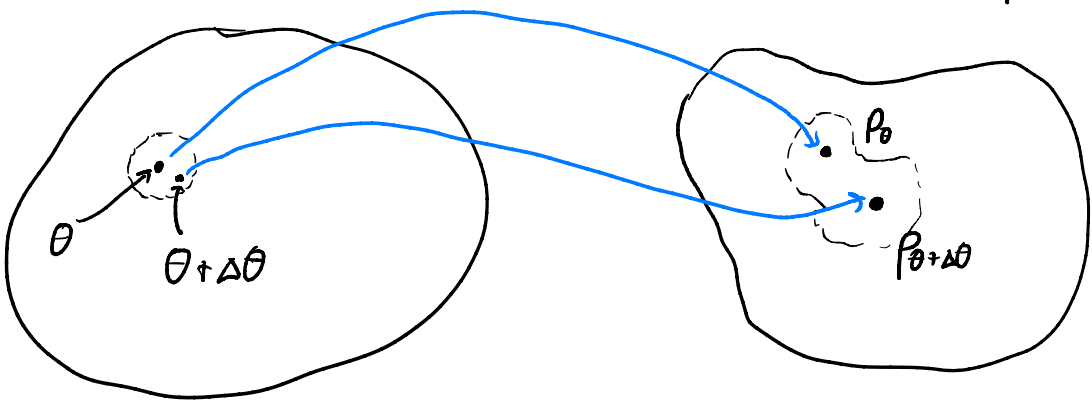
$\Leftrightarrow \; -\log r = \ell(\text{CrossEntropyLoss})$

$\Rightarrow$ The NN models a Gaussian/categorical distribution

Probabilistic perspective

parameter space

statistical manifold



$$d(\theta, \theta+\Delta\theta) = \|\Delta\theta\|_2^2$$

$$d(p_{\theta+\Delta\theta}, p_\theta)$$

$$= \left(\begin{array}{l}\text{second-order approximation of} \\ KL(p_{\theta+\Delta\theta} \| p_\theta)\end{array}\right)$$

$$= -\frac{1}{2} \mathbb{E}_{p_\theta(x,y)} \left[\nabla_\theta^2 \log p_\theta(x,y) \right]$$

"Fisher information matrix"

(use $p_\theta(x,y) = p_\theta(y|x)\, p_{\mathcal{D}}(x)$)

$$F(\theta) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{p_\theta(y|x_n)} \left[-\nabla_\theta^2 \log p_\theta(y|x_n) \right]$$

$$- \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{p_\theta(y|x_n)} \left[-\nabla_\theta^2 \ell(f(x_n,\theta), y) \right]$$

looks similar to Hessian

$$= \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{J}_\theta f_n\right)^\top \mathbb{E}_{p_\theta(y|x_n)} \left[\nabla_f^2 \ell(f_n, y) \right] \mathbf{J}_\theta f_n$$

Type-II looks like GGN

$$= \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{J}_\theta f_n\right)^\top \mathbb{E}_{p_\theta(y|x_n)} \left[\nabla_f \ell(f_n, y) \left(\nabla_f \ell(f_n, y)\right)^\top \right] \mathbf{J}_\theta f_n$$

Type-I motivation for EF

Similarities and differences $(\ell = -\log p)$

$$H(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x,y)}\left[\nabla_\theta^2 \ell\right] = \mathbb{E}_{p_{\mathcal{D}}(x)}\mathbb{E}_{p_{\mathcal{D}}(y|x)}\left[\nabla_\theta^2 \ell\right]$$ Hessian

$$F(\theta) = \mathbb{E}_{p_\theta(x,y)}\left[\nabla_\theta^2 \ell\right]$$ Fisher

$$F_{II}(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x)}\left[(J_\theta f)^T \mathbb{E}_{p_\theta(y|x)}\left[\nabla_f^2 \ell\right] J_\theta f\right]$$ Fisher type-II

$$G(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x)}\left[(J_\theta f)^T \mathbb{E}_{p_{\mathcal{D}}(y|x)}\left[\nabla_f^2 \ell\right] J_\theta f\right]$$ GGN

$$F_I(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x)}\left[(J_\theta f)^T \mathbb{E}_{p_\theta(y|x)}\left[\nabla_f \ell\, (\nabla_f \ell)^T\right] J_\theta f\right]$$ Fisher type-I

$$EF(\theta) = \mathbb{E}_{p_{\mathcal{D}}(x)}\left[(J_\theta f)^T \mathbb{E}_{p_{\mathcal{D}}(y|x)}\left[\nabla_f \ell\, (\nabla_f \ell)^T\right] J_\theta f\right]$$ empirical Fisher

Matrix-free linear algebra with linear operators

Hutchinson trace/diagonal estimation

Trace

$$\mathrm{Tr}(A) = \mathrm{Tr}(AI)$$

$$= \mathrm{Tr}\left(A\, \mathbb{E}_{p(v)}[vv^T]\right)$$

$$= \mathbb{E}_{p(v)}\left[\mathrm{Tr}(Avv^T)\right]$$

$$= \mathbb{E}_{p(v)}\left[\mathrm{Tr}(v^TAv)\right]$$

$$= \mathbb{E}_{p(v)}\left[v^TAv\right]$$

$$\approx \frac{1}{S}\sum_{s=1}^{S} v_s^T A v_s \quad \text{where } v_s \overset{\text{i.i.d.}}{\sim} p(v)$$

(can also use to estimate $\|A\|_2^2 = \mathrm{Tr}(A^TA)$)

($\mathbb{E}_{p(v)}[vv^T] = I$, e.g. zero mean unit covariance Gaussian or Rademacher)

//

Diagonal

$$\mathrm{diag}(A) = \mathrm{diag}(AI)$$

$$= \mathrm{diag}\left(A\, \mathbb{E}_{p(v)}[vv^T]\right)$$

$$= \mathbb{E}_{p(v)}\left[\mathrm{diag}(Avv^T)\right]$$

$$= \mathbb{E}_{p(v)}\left[(Av) \odot v\right]$$

$$\approx \frac{1}{S}\sum_{s=1}^{S} Av_s \odot v_s \quad \text{where } v_s \overset{\text{i.i.d.}}{\sim} p(v)$$

//

Power iteration

eig

$\lambda_1(A)$, $e_1(A)$

$\lambda_1 \leftarrow \infty$
$e_1 \leftarrow v$ — v random, $\|v\|_2 = 1$

until converged

- $e_1 \leftarrow A e_1$ — matvec
- $\lambda_1 \leftarrow \|e_1\|_2$ — update eigval
- $e_1 \leftarrow e_1 / \|e_1\|_2$

- (more sophisticated methods, e.g. implicitly restarted Arnoldi iterations (ARPACK), scipy.sparse.linalg.eigsh)

- Also: Spectral density estimation

$$\rho(\lambda) = \frac{1}{D} \sum_{d=1}^{D} \delta(\lambda - \lambda_d)$$

//

Linear systems and inversion

linsolve $(A, b) = x$ such that $Ax = b$.

↳ Iterative solvers like conjugate gradients (CG) (details not important, only uses $[v \mapsto Av]$).

//

CG inversion: Compute $A^{-1}b = x$

$\Leftrightarrow$ Find x : $b = Ax$

$\Rightarrow x = \text{linsolve}(A, b)$

//

Neumann series:

$$A^{-1} = \sum_{k=0}^{\infty} (I - A)^k \qquad \left(\text{if } 0 < \lambda(A) < 2\right)$$

$$A^{-1}v = v + (I-A)v + (I-A)^2 v + \dots \qquad (\text{truncate})$$

curvlinops library: offers linear operators for H, $F_{II}, \tilde{F}_{I}, G$, EF, and more + functionality to estimate their properties