Saddle Points

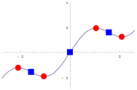
Elena Burceanu
eburceanu@bitdefender.com

Dauphin and Pascanu

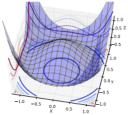
- Identifying and attacking the saddle point problem in high-dimensional non-convex optimization - Pascanu et al. [2014] and Dauphin et al. [2014]
- prior work about the geometry of the error function
- optimization algorithms behavior (near saddle points)
- new algorithm (Saddle Free)
- practical implementation of
  - SGD (ok, but slow)
  - Newton (doesn't escape from SPs)
  - Natural Gradient (might not escape)
  - Saddle-Free Newton (new solution)

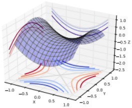
Definitions

- Stationary (critical) point
  - $\forall i, \frac{\partial F}{\partial \theta_i}(\theta_0) = 0$

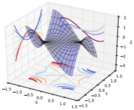


- Point of inflexion
  - $F"(\theta_0) = 0$ (defined only in 1D)
- Minima (maxima)
  - stationary point
  - Hessian matrix analysis
  - min: $\forall i, \lambda_i \geq 0$ (max: $\forall i, \lambda_i \leq 0$)





- Saddle point
  - stationary point
  - not a local minima/maxima
  - analyze Hessian matrix in θ_0
  - $\exists i, j (i \neq j)$ s.t. $\lambda_i > 0$ and $\lambda_j < 0$
  - degenerates (monkey SP) $\exists i, \lambda_i = 0$



Generic cost landscapes

- Bray and Dean [2007], Fyodorov and Williams [2007]
  - statistical physics
  - Gaussian random matrix
    - replica theory
    - w - stationary points, ϵ - $Error(w_0)$
    - α - % negative eigenvalues of the Hessian(w_0)
    - monotonically increasing curve: "the larger the error, the larger the index"
    - if w_i is a local minima, then $\alpha_i = 0$, so ϵ_i is close to global minima
    - if ϵ_i is large, then $\alpha_i > 0$, so w_i is a saddle point
    - Wigners famous semicircular law
    - spectrum is shifting right

Generic cost landscapes

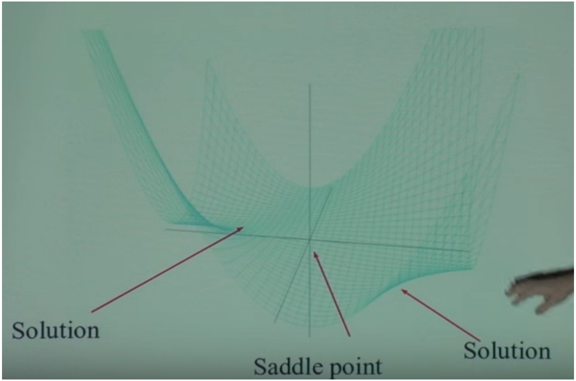
- Bray and Dean [2007], Fyodorov and Williams [2007]
  - statistical physics
  - Gaussian random matrix
    - replica theory
    - w - stationary points, ϵ - $Error(w_0)$
    - α - % negative eigenvalues of the Hessian(w_0)
    - monotonically increasing curve: "the larger the error, the larger the index"
    - if w_i is a local minima, then $\alpha_i = 0$, so ϵ_i is close to global minima
    - if ϵ_i is large, then $\alpha_i > 0$, so w_i is a saddle point
    - Wigners famous semicircular law
    - spectrum is shifting right
  - random matrix theory
    - $P(\lambda_i > 0) = \frac{1}{2}$
    - $P(\lambda_i > 0) = (\frac{1}{2})^{N_\lambda}, \forall i$ $(1..N_\lambda)$, discussion over N_λ

Cost function - Practical landscape

- downsample input, compute J, H, eigenvalues
- Baldi and Hornik [1989]
  - 1 layer MLP, linear
  - error surface shows only saddle-points and no local minima
- Saxe et al. [2014]
  - linear MLP
  - SP arise due to scaling symmetries in the weight space (Jacobians isometry)
  - orthogonal weight initialization $\rightarrow$ training time DOESN'T depend on MLP depth
  - linear nets have many saddle points
- Mizutani and Dreyfus [2010]
  - 1 layer MLP
  - error surface has saddle points (where the Hessian matrix is indefinite)

Symmetries in error landscapes

- Rattray et al. [1998], Inoue et al. [2003]
  - symmetries in error function: $F(\theta^{(1)}) = F(\theta^{(2)})$
  - going from $\theta^{(1)}$ to $\theta^{(2)}$ should pass over a saddle point (frequent) or a local minima/maxima (very rare)



Quick review of GD and Newton optimization

- f is 2-differentiable, convex on a convex subset $\leftrightarrow$ Hessian is positive semidefinite on that subset
- Taylor: $f(\theta_0 + p) \approx f(\theta_0) + p^T \nabla_\theta f(\theta_0) + \frac{p^T H_f(\theta_0) p}{2}$
- find p that minimize f (near θ_0)
- **Gradient Descent**
  - fix step size ($||p||_2 = 1$)
  - $f(\theta_0 + \alpha p) = const + \alpha p^T \nabla_\theta f(\theta_0)$, α is small
  - **Q: p = ?**

Quick review of GD and Newton optimization

- f is 2-differentiable, convex on a convex subset $\leftrightarrow$ Hessian is positive semidefinite on that subset
- Taylor: $f(\theta_0 + p) \approx f(\theta_0) + p^T \nabla_\theta f(\theta_0) + \frac{p^T H_f(\theta_0) p}{2}$
- find p that minimize f (near θ_0)
- **Gradient Descent**
  - fix step size ($||p||_2 = 1$)
  - $f(\theta_0 + \alpha p) = const + \alpha p^T \nabla_\theta f(\theta_0)$, α is small
  - **Q: p = ?**
  - $minimize_p : p^T \nabla_\theta f(\theta_0) = ||p^T||_2 * ||\nabla_\theta f(\theta_0)||_2 * cos(\beta)$
  - solution: $cos(\beta) = -1$, $p = -\frac{\nabla_\theta f(\theta_0)}{||\nabla_\theta f(\theta_0)||_2}$, iterate
- **Newton**
  - second order Taylor approximation
  - **Q: p = ?**

Quick review of GD and Newton optimization

- f is 2-differentiable, convex on a convex subset $\leftrightarrow$ Hessian is positive semidefinite on that subset
- Taylor: $f(\theta_0 + p) \approx f(\theta_0) + p^T \nabla_\theta f(\theta_0) + \frac{p^T H_f(\theta_0) p}{2}$
- find p that minimize f (near θ_0)
- **Gradient Descent**
  - fix step size ($||p||_2 = 1$)
  - $f(\theta_0 + \alpha p) = const + \alpha p^T \nabla_\theta f(\theta_0)$, α is small
  - **Q: p = ?**
  - $minimize_p : p^T \nabla_\theta f(\theta_0) = ||p^T||_2 * ||\nabla_\theta f(\theta_0)||_2 * cos(\beta)$
  - solution: $cos(\beta) = -1$, $p = -\frac{\nabla_\theta f(\theta_0)}{||\nabla_\theta f(\theta_0)||_2}$, iterate
- **Newton**
  - second order Taylor approximation
  - **Q: p = ?**
  - condition: $\frac{\partial f(\theta_0 + p)}{\partial p} = 0$
  - solve: $\nabla_\theta f(\theta_0)^T + \frac{p^T (H_f(\theta_0) + H_f(\theta_0)^T)}{2} = 0$
  - solution: $p = -H_f(\theta_0)^{-1} * \nabla_\theta f(\theta_0)$
- find more in Nocedal and Wright [2006a]

Optimization near Saddle Points

- SP are very frequent
- how optimization algorithms behave near them?
- θ^* is a critical point: $\forall i, \frac{\partial f(\theta)}{\partial \theta_i}(\theta^*) = 0$
- Taylor second order approximation near θ^* - SP
  - $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{\Delta\theta^T H_f(\theta^*)\Delta\theta}{2}$
  - $H = H^T = VDV^T, V = [v_1|v_2|...]$ **Q: Why?**

Optimization near Saddle Points

- SP are very frequent
- how optimization algorithms behave near them?
- θ^* is a critical point: $\forall i, \frac{\partial f(\theta)}{\partial \theta_i}(\theta^*) = 0$
- Taylor second order approximation near θ^* - SP
  - $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{\Delta\theta^T H_f(\theta^*) \Delta\theta}{2}$
  - $H = H^T = VDV^T, V = [v_1|v_2|...]$ **Q: Why?**
  - Spectral theorem
  - $H = \sum_{i=1}^{n} \lambda_i v_i v_i^T$, $H^{-1} = \sum_{i=1}^{n} \lambda_i^{-1} v_i v_i^T$

Optimization near Saddle Points

- SP are very frequent
- how optimization algorithms behave near them?
- θ^* is a critical point: $\forall i, \frac{\partial f(\theta)}{\partial \theta_i}(\theta^*) = 0$
- Taylor second order approximation near θ^* - SP
  - $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{\Delta\theta^T H_f(\theta^*)\Delta\theta}{2}$
  - $H = H^T = VDV^T, V = [v_1|v_2|...]$ **Q: Why?**
  - Spectral theorem
  - $H = \sum_{i=1}^n \lambda_i v_i v_i^T$, $H^{-1} = \sum_{i=1}^n \lambda_i^{-1} v_i v_i^T$
  - $\Delta\theta^T H \Delta\theta = \Delta\theta^T (\sum_{i=1}^n \lambda_i v_i v_i^T)\Delta\theta = \sum_{i=1}^n \lambda_i (v_i^T \Delta\theta)^2$
  - $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2} * \sum_{i=1}^n \lambda_i (v_i^T \Delta\theta)^2$
- optimization algorithms: find next $\Delta\theta$
- how good minimizer are $\Delta\theta$ for f?

A. Gradient Descent near Saddle Points

Bitdefender

- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2}\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- $stepSGD = -\nabla_\theta f(\theta^* + \Delta\theta) = -\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta) v_i^T$
- $stepSGD_{v_i} = -\lambda_i (v_i^T \Delta\theta)$
- $\Delta\theta = \sum_{j=1}^{n} \epsilon_j v_j$ (**Q: Why do** $v_j s$ **form a basis?**)

A. Gradient Descent near Saddle Points

- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2}\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- $stepSGD = -\nabla_\theta f(\theta^* + \Delta\theta) = -\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta) v_i^T$
- $stepSGD_{v_i} = -\lambda_i (v_i^T \Delta\theta)$
- $\Delta\theta = \sum_{j=1}^{n} \epsilon_j v_j$ (**Q: Why do** $v_j s$ **form a basis?**)
- $v_i^T \Delta\theta = v_i^T \sum_j \epsilon_j v_j = \epsilon_i \rightarrow stepSGD_{v_i} = -\lambda_i \epsilon_i$
- update rule: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - \alpha\lambda_i)\epsilon_i * v_i$
- **Q: Analysis over** $\lambda_i < 0$, $\lambda_j > 0$

A. Gradient Descent near Saddle Points

- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2}\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- $stepSGD = -\nabla_\theta f(\theta^* + \Delta\theta) = -\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta) v_i^T$
- $stepSGD_{v_i} = -\lambda_i (v_i^T \Delta\theta)$
- $\Delta\theta = \sum_{j=1}^{n} \epsilon_j v_j$ (**Q: Why do $v_j s$ form a basis?**)
- $v_i^T \Delta\theta = v_i^T \sum_j \epsilon_j v_j = \epsilon_i \rightarrow stepSGD_{v_i} = -\lambda_i \epsilon_i$
- update rule: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - \alpha\lambda_i)\epsilon_i * v_i$
- **Q: Analysis over $\lambda_i < 0$, $\lambda_j > 0$**
  - moves away from θ^*, in v_i (negative curvature) direction
  - moves towards θ^*, in v_j (positive curvature) direction
  - BUT proportionally with λ_i value
  - for a large discrepancy between eigenvalues, GD can be very slow
- GD (slowly) escapes SP

B. Newton near Saddle Points

- Newton assumption: Hessian is positive definite
- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2}\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- stepNewton $= -H_f^{-1} * \nabla_\theta f$
- $-(\sum_{i=1}^{n} \lambda_i^{-1} v_i v_i^T)(\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta) v_i^T)^T = -\sum_{i=1}^{n} (v_i^T \Delta\theta) v_i$
- $stepNewton_{v_i} = -v_i^T \Delta\theta = -\epsilon_i$
- update rule: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1-1)\epsilon_i * v_i$
- **Q: Is it bad, is it good?**

B. Newton near Saddle Points

- Newton assumption: Hessian is positive definite
- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2}\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- stepNewton $= -H_f^{-1} * \nabla_\theta f$
- $-(\sum_{i=1}^{n} \lambda_i^{-1} v_i v_i^T)(\sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta) v_i^T)^T = -\sum_{i=1}^{n}(v_i^T \Delta\theta) v_i$
- $stepNewton_{v_i} = -v_i^T \Delta\theta = -\epsilon_i$
- update rule: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n}(1-1)\epsilon_i * v_i$
- **Q: Is it bad, is it good?**
- losses info about $sign(\lambda_i)$, SPs on any direction
- Newton DOESN'T escape SP; it is a SP attractor

C. Hessian approximation

- practical implementation of 2^{nd} order methods for non-convex optimization (trust region)
- non-convex; Hessian has negative curvature ($\lambda_i < 0$)
- currently, we ignore the negative curvature (we suppose that the problem is convex)
- damping the Hessian (to remove the negative curvature) $H = VD_{damped}V^T; D_{damped} = D + m * \mathbb{I}, H \leftarrow H + m * \mathbb{I}$
  - m min (we want small change in H) s.t. $\lambda_{min} + m > 0$
- $stepTR = -H_{damped}^{-1} * \nabla_\theta f$
- $stepTR_{v_i} = -\frac{\lambda_i}{\lambda_i + m} v_i^T \Delta\theta = -\frac{\lambda_i}{\lambda_i + m}\epsilon_i$
- **Q: Is it all fixed? Discuss this result.**

C. Hessian approximation

Bitdefender

- practical implementation of 2^{nd} order methods for non-convex optimization (trust region)
- non-convex; Hessian has negative curvature ($\lambda_i < 0$)
- currently, we ignore the negative curvature (we suppose that the problem is convex)
- damping the Hessian (to remove the negative curvature) $H = VD_{damped}V^T; D_{damped} = D + m * \mathbb{I}, H \leftarrow H + m * \mathbb{I}$
  - m min (we want small change in H) s.t. $\lambda_{min} + m > 0$
- $stepTR = -H_{damped}^{-1} * \nabla_\theta f$
- $stepTR_{v_i} = -\frac{\lambda_i}{\lambda_i+m} v_i^T \Delta\theta = -\frac{\lambda_i}{\lambda_i+m}\epsilon_i$
- **Q: Is it all fixed? Discuss this result.**
- same problem as GD, for a large discrepancy between eigenvalues, adding a fix m to each λ_i might reduce $\frac{\lambda_i}{\lambda_i+m}$ very close to 0 (for some i); slow

D: Natural Gradient near Saddle Points (opt)

- **Q: Linear search vs Trust Region?**

D: Natural Gradient near Saddle Points (opt)

- **Q: Linear search vs Trust Region?**
- Trust region: $\text{argmin}_{\Delta\theta} f(\theta + \Delta\theta)$, s.t. $KL(p_\theta || p_{\theta+\Delta\theta}) < \epsilon$
- 2^{nd} order Taylor approximation: $KL(p_\theta || p_{\theta+\Delta\theta}) = \frac{1}{2}\Delta\theta^T F \Delta\theta$ (Berkeley CS 287: Advanced Robotics)
- Fisher matrix is a first order approximation for the Hessian and it is **positive definite** $F = -E[H]$ (see Appendix)
- $stepNG = -F^{-1} * \nabla_\theta f$
- **Q: Where does this formula came from?**

D: Natural Gradient near Saddle Points (opt)

- **Q: Linear search vs Trust Region?**
- Trust region: $\text{argmin}_{\Delta\theta} f(\theta + \Delta\theta)$, s.t. $KL(p_\theta || p_{\theta+\Delta\theta}) < \epsilon$
- 2^{nd} order Taylor approximation: $KL(p_\theta || p_{\theta+\Delta\theta}) = \frac{1}{2}\Delta\theta^T F \Delta\theta$ (Berkeley CS 287: Advanced Robotics)
- Fisher matrix is a first order approximation for the Hessian and it is **positive definite** $F = -E[H]$ (see Appendix)
- $stepNG = -F^{-1} * \nabla_\theta f$
- **Q: Where does this formula came from?**
- near SP, $H(\theta^*) - E[H(\theta^*)]$ might be too big
- other reasons: Mizutani and Dreyfus [2010] (related to the singularity of F)
- Natural Gradient might NOT escape SP

E: Saddle free algorithm

- Trust Region approach
  - $\arg\min_{\Delta\theta}$ *TaylorAprox*$_k f(\theta + \Delta\theta)$ for a value of $k = 1, 2$
  - s. t. $d(\theta, \theta + \Delta\theta) \leq \Delta$
- Saddle free algorithm (intuition)
  - simple idea, based on previous observations:
    - step should depend on $sign(\lambda_i)$
    - step should NOT depend on $|\lambda_i|$
  - **Q: How should the step (and H) look like?**

E: Saddle free algorithm

- Trust Region approach
  - $\arg\min_{\Delta\theta} TaylorAprox_k f(\theta + \Delta\theta)$ for a value of $k = 1, 2$
  - s. t. $d(\theta, \theta + \Delta\theta) \leq \Delta$
- Saddle free algorithm (intuition)
  - simple idea, based on previous observations:
    - step should depend on $sign(\lambda_i)$
    - step should NOT depend on $|\lambda_i|$
  - **Q: How should the step (and H) look like?**
  - step rescaled with $\frac{1}{|\lambda_i|}$
  - new Hessian: $|H| = V|D|V^T; H^{-1} = V|D|^{-1}V^T$
  - $|D|$ has absolute values of eigenvalues instead of simple eigenvalues
  - idea was mentioned, without proof: Nocedal and Wright [2006b] or in Murray [2010]
- Saddle free algorithm (formal)
  - $\Delta\theta_{SFA} = \arg\min_{\Delta\theta} f(\theta) + \Delta\theta^T \nabla_\theta f(\theta)$
  - how far from θ can we trust the first order approx?
  - $d(\theta, \theta + \Delta\theta) = |TaylorAprox_2 - TaylorAprox_1|$
  - $d(\theta, \theta + \Delta\theta) = \frac{1}{2}|\Delta\theta^T H \Delta\theta| \leq \frac{1}{2}\Delta\theta^T |H| \Delta\theta \leq \Delta$
  - Lagrange multipliers: $stepSF = -|H|^{-1} * \nabla_\theta f$

Recap

- $f(\theta^* + \Delta\theta) = f(\theta^*) + \frac{1}{2} * \sum_{i=1}^{n} \lambda_i (v_i^T \Delta\theta)^2$
- $\Delta\theta = \sum_{i=1}^{n} \epsilon_i v_i$
- $v_i^T \Delta\theta = \epsilon_i$
- $\theta_{new} \leftarrow \theta_{old} - \alpha * step$
- **SGD**:$\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - \alpha\lambda_i)\epsilon_i * v_i$
- **Newton**: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - 1)\epsilon_i * v_i$
- **damped Hessian**: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - \frac{\lambda_i}{\lambda_i + m})\epsilon_i * v_i$
- **Saddle Free**: $\theta_{new} \leftarrow \theta^* + \sum_{i=1}^{n} (1 - \frac{\lambda_i}{|\lambda_i|})\epsilon_i * v_i$
- Wanted behavior
  - $\lambda_i > 0$, want to go closer to the SP (is the minimum on this subspace)
  - $\lambda_i < 0$, want to go further from the SP (is maximum on this subspace)

Experiments

- Practical implementation problems
  - hard to compute Hessian (n X n, too large for many parameters)
  - hard to inverse Hessian
  - **Q: How would you implement it?**

Experiments

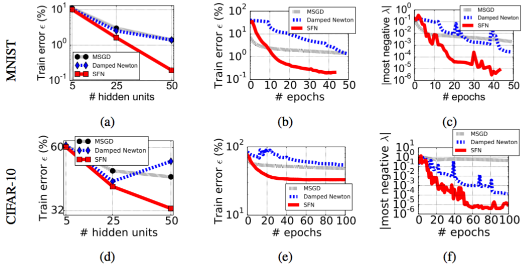
- Practical implementation problems
  - hard to compute Hessian (n X n, too large for many parameters)
  - hard to inverse Hessian
  - **Q: How would you implement it?**
  - see Appendix 2.
- Results
  - MNIST and CIFAR-10, 10x10 downsampled
  - 7 layers deep MLP; RNN on Penn Treebank
  - optimization: SGD first, continue with SFA
  - eigenvalues distribution shifts right
  - SFA vs other algo: better for more parameters

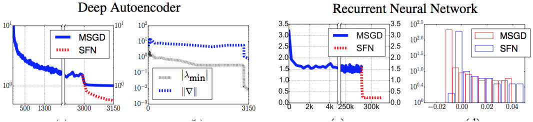
Statistically relevant experiments

- critical points distribution in the $\epsilon - \alpha$ plane
- how the eigenvalues of the Hessian at these critical points are distributed
- MNIST downsampled
  - along optimization path, find nearby critical points
  - (Newton's method: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$)
  - 20 runs of SFA (random seed)
  - 100 jobs - find critical points around parameters from random epochs (of 20 random SFA runs)
  - 100 jobs - find critical points with random sampling (-1, 1)
- CIFAR downsampled
  - 3 layer NN, SGD, tanh, 10-300 epochs, random init $\rightarrow$ save all params
  - Newton's method
- results (confirms Bray and Dean [2007]):
  - eigenvalues distribution shift to the left as the error increases
  - critical points concentrate along a monotonically increasing curve in the $\epsilon - \alpha$ plane

Results

- **Q: Something interesting?**





Conclusion

- Theoretical existence of Saddle Points (others)
- Practical existence of Saddle Points
  - (statistically) relevant experiments
- Optimization algorithms: behavior near SP
- New algorithm (Saddle Free algorithm)
  - demonstration
  - practical implementation, difficulties, framework
- Future work
  - better H estimation algorithms
  - find new theoretical properties of SP in NN context, understand statistical property of high dimensional surfaces
- "Saddle-free Hessian-free Optimization", Martin Arjovsky NYU, workshop NIPS 2016

Other Questions?

Video about the subject (introduction): Bengio [2015]

Appendix 1A: Fisher Matrix

Bitdefender

- the amount of info that X (observable random variable) carries about θ (unknown parameter)
- $f(X|\theta) = f_\theta(X)$ probability for X, likelihood for θ
- score $= \frac{\partial \log f_\theta(X)}{\partial\theta}$
- $E_{f_\theta(X)}[score] = 0$ (first moment)
  - $= E_{f_\theta(X)}[\frac{\partial \log f_\theta(X)}{\partial\theta}] = E_{f_\theta(X)}[\frac{\partial \log f_\theta(X)}{\partial f_\theta(X)} \frac{\partial f_\theta(X)}{\partial\theta}] = E_{f_\theta(X)}[\frac{1}{f_\theta(X)} \frac{\partial f_\theta(X)}{\partial\theta}] = \int \frac{1}{f_\theta(X)} \frac{\partial f_\theta(X)}{\partial\theta} f_\theta(X) dx = \int \frac{\partial f_\theta(X)}{\partial\theta} dx = \frac{\partial}{\partial\theta} \int f_\theta(X) dx = \frac{\partial 1}{\partial\theta} = 0$
- $E_{f_\theta(X)}[score^2]$ (second moment = Fisher info)
  - $H = \frac{\partial^2 \log f_\theta(X)}{\partial\theta^2} = \frac{\partial}{\partial\theta}\frac{\partial \log f_\theta(X)}{\partial\theta} = \frac{\partial}{\partial\theta}\frac{\frac{\partial f_\theta(X)}{\partial\theta}}{f_\theta(X)} = \frac{g' * h - h' * g}{h^2} = \frac{\frac{\partial^2 f_\theta(X)}{\partial\theta^2}}{f_\theta(X)} - (\frac{\frac{\partial f_\theta(X)}{\partial\theta}}{f_\theta})^2 = \frac{\frac{\partial^2 f_\theta(X)}{\partial\theta^2}}{f_\theta(X)} - (\frac{\partial \log f_\theta(X)}{\partial\theta})^2$
  - $\int \frac{\frac{\partial^2 f_\theta(X)}{\partial\theta^2}}{f_\theta(X)} f_\theta(X) dx = 0$
  - $E[H] = -\int (\frac{\partial \log f_\theta(X)}{\partial\theta})^2 f_\theta(X) dx = -FisherMatrix$

Appendix 1B: Natural Gradient Learning in MLP

- Amari [1998]
- $q(x)$ = real distribution; $p_\theta(x) = estimate$; find θ which approximates it best
- $Loss = -E_q[log p_\theta(x)] = E_q[log \frac{q(x)}{q(x)} - log p_\theta(x)] =$
  $E_q[log \frac{q(x)}{p_\theta(x)}] - E_q[log q(x)] = E[log \frac{q(x)}{p_\theta(x)}] + entropy_q$
- $Loss(\theta) = KL(q||p_\theta) + const.$
- 2^{nd} order Taylor approx: $KL(p_\theta||p_{\theta+\Delta\theta}) = \frac{1}{2}\Delta\theta^T F \Delta\theta$ (Berkeley CS 287: Advanced Robotics)
- **Q: Why is the Fisher matrix important? Demonstrate that $\lambda_i > 0, \forall i$**

Appendix 1B: Natural Gradient Learning in MLP

- Amari [1998]
- $q(x)$ = real distribution; $p_\theta(x) = estimate$; find θ which approximates it best
- $Loss = -E_q[log p_\theta(x)] = E_q[log\frac{q(x)}{q(x)} - log p_\theta(x)] = E_q[log\frac{q(x)}{p_\theta(x)}] - E_q[log q(x)] = E[log\frac{q(x)}{p_\theta(x)}] + entropy_q$
- $Loss(\theta) = KL(q||p_\theta) + const.$
- 2^{nd} order Taylor approx: $KL(p_\theta||p_{\theta+\Delta\theta}) = \frac{1}{2}\Delta\theta^T F \Delta\theta$ (Berkeley CS 287: Advanced Robotics)
- **Q: Why is the Fisher matrix important? Demonstrate that** $\lambda_i > 0, \forall i$
- $x^T * F * x = E[X^T * S * S^T * X)] = E[(X^T * S)^2] \geq 0$

Appendix 2A: Power Iteration (PageRank)

- given A, the algo finds the biggest λ_i and its eigenvector
- $\frac{A^k x}{||A^k x||_2} \rightarrow_k v_1^*$ (principal eigenvector)
- $A = VJV^{-1} \Rightarrow A^k = VJ^kV^{-1}$ (Jordan decomposition)
- $x = \sum_{i=1}^n c_i v_i$, random vector x (v_i form a base)
- $A^k x = VJ^kV^{-1}(\sum_{i=1}^n c_i v_i) =$
  $VJ^kV^{-1}c_1 v_1 + VJ^kV^{-1}(\sum_{i=2}^n c_i v_i)$
- $A^k x = \lambda_1^k c_1 v_1 + \lambda_1^k V(\frac{J}{\lambda_1})^k(\sum_{i=2}^n c_i e_i)$
- $(\frac{J}{\lambda_1})^k =_{k\rightarrow\infty} \begin{bmatrix} 1 & 0 & 0 & \dots \\ 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \Rightarrow (\frac{J}{\lambda_1})^k e_i = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dots \end{bmatrix}, i \geq 2$
- convergence rate: $(\frac{J}{\lambda_1})^k$ converges geometrical with $(\frac{\lambda_2}{\lambda_1})$ rate
- $||A^k x||_2 = \lambda_1^k c_1$, $\frac{A^k x}{||A^k x||_2} \rightarrow v_1^*$ (iterative, no decomposition)

Appendix 2B: Lanczos algorithm

- in Power Iteration (PI), $x, Ax, A^2x, ...$ become linear dependent
- PI is numeric instable
- orthogonalized base for faster convergence
- Krylov subspace $x, Ax, A^2x, ...$
- PI throws away previous computation
- make the base orthogonal $u_i = v_i - \sum_{k=1}^{i} proj_{u_k} v_i$ (Gram Schmidt)
- normalize the base $\frac{u_i}{||u_i||_2}$
- Lanczos algo
  - compute new vector: $(w_i = Hv_i)$
  - apply Gram Schmidt for w_i to make the basis orthogonal
  - normalize $v_{i+1} = \frac{w_i}{||w_i||_2}$
- easy to compute the inverse of a matrix, having the Krylov space (linear combination of its powers)

References I

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276, 1998. URL http://www.maths.tcd.ie/~mnl/store/Amari1998a.pdf.

P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989. doi: 10.1016/0893-6080(89)90014-2. URL http://dx.doi.org/10.1016/0893-6080(89)90014-2.

Y. Bengio. Deep learning: Theoretical motivations, 2015. URL http://videolectures.net/deeplearning2015_bengio_theoretical_motivations/.

c. a. N. G. Berkeley CS 287: Advanced Robotics. Berkeley - cs 287: Advanced robotics, course about natural gradient. URL https://people.eecs.berkeley.edu/~pabbeel/cs287-fa09/lecture-notes/lecture20-6pp.pdf.

References II

A. J. Bray and D. S. Dean. Statistics of critical points of gaussian fields on large-dimensional spaces. *Physics Review Letter*, 2007. URL `https://arxiv.org/abs/cond-mat/0611023`.

Y. Dauphin, R. Pascanu, Ç. Gülçehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *CoRR*, abs/1406.2572, 2014. URL `http://arxiv.org/abs/1406.2572`.

Y. V. Fyodorov and I. Williams. Replica symmetry breaking condition exposed by random matrix calculation of landscape complexity. *Journal of Statistical Physics*, 2007. URL `https://arxiv.org/abs/cond-mat/0702601`.

References III

M. Inoue, H. Park, and M. Okada. On-line learning theory of soft committee machines with correlated hidden units steepest gradient descent and natural gradient descent. *Journal of the Physical Society of Japan*, 72(4):805–810, 2003. doi: 10.1143/JPSJ.72.805. URL `http://dx.doi.org/10.1143/JPSJ.72.805`.

E. Mizutani and S. Dreyfus. An analysis on negative curvature induced by singularity in multi-layer neural-network learning. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 1669–1677, 2010. URL `http://papers.nips.cc/paper/4046-an-analysis-on-negative-curvature-induced-by-singul`

W. Murray. Newton-type methods, 2010.

References IV

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006a.

J. Nocedal and S. J. Wright. *Numerical Optimization*. 2006b.

R. Pascanu, Y. N. Dauphin, S. Ganguli, and Y. Bengio. On the saddle point problem for non-convex optimization. *CoRR*, abs/1405.4604, 2014. URL `http://arxiv.org/abs/1405.4604`. Explain better.

M. Rattray, D. Saad, and S.-i. Amari. Natural gradient descent for on-line learning. *Phys. Rev. Lett.*, 81:5461–5464, Dec 1998. doi: 10.1103/PhysRevLett.81.5461. URL `http://link.aps.org/doi/10.1103/PhysRevLett.81.5461`.

A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *CoRR*, abs/1312.6120, 2014. URL `http://arxiv.org/abs/1312.6120`.