Lecture 5: Training Deep Neural Networks II

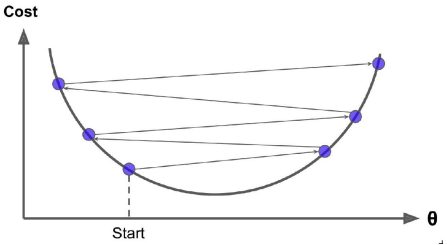
Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Advanced Machine Learning
September–December 2021

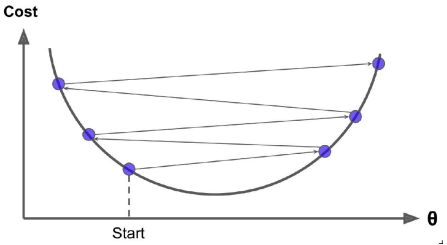
Ill conditioning

- Ill conditioning — small change in input produces a large change in output



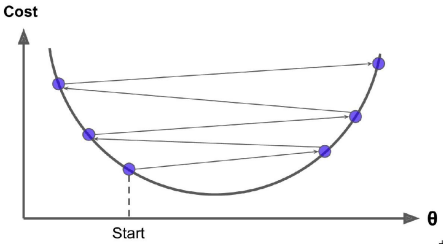
Ill conditioning

- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \dfrac{\partial}{\partial \theta_i} J(\theta)$



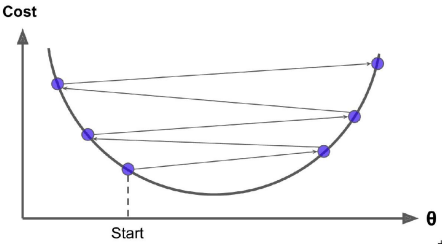
Ill conditioning

- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \dfrac{\partial}{\partial \theta_i} J(\theta)$
- Impact of update $\theta - \epsilon \nabla_\theta$ on cost $J(\theta)$?



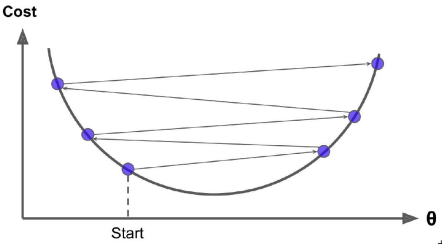
Ill conditioning

- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \frac{\partial}{\partial \theta_i} J(\theta)$
- Impact of update $\theta - \epsilon \nabla_\theta$ on cost $J(\theta)$?
- Depends on curvature, given by second derivative



Ill conditioning

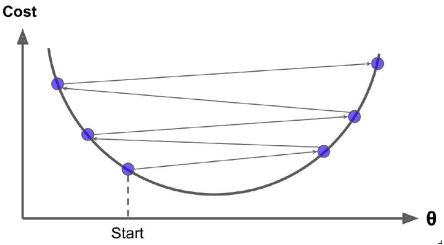
- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \frac{\partial}{\partial \theta_i} J(\theta)$
- Impact of update $\theta - \epsilon \nabla_\theta$ on cost $J(\theta)$?
- Depends on curvature, given by second derivative
- Hessian: $H_\theta = \frac{\delta^2}{\delta \theta_i \delta \theta_j} J(\theta)$



Ill conditioning

- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \frac{\partial}{\partial \theta_i} J(\theta)$
- Impact of update $\theta - \epsilon \nabla_\theta$ on cost $J(\theta)$?
- Depends on curvature, given by second derivative
- Hessian: $H_\theta = \frac{\delta^2}{\delta \theta_i \delta \theta_j} J(\theta)$
- Using Taylor expansion, impact of update $\theta - \epsilon \nabla_\theta$,

$$J(\theta) - \nabla_\theta^T \nabla_\theta + \frac{1}{2} \nabla_\theta^T H_\theta \nabla_\theta$$

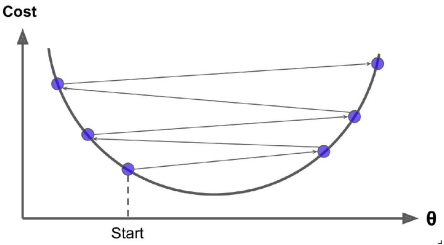


Ill conditioning

- Ill conditioning — small change in input produces a large change in output
- Gradient $\nabla_\theta = \frac{\partial}{\partial \theta_i} J(\theta)$
- Impact of update $\theta - \epsilon \nabla_\theta$ on cost $J(\theta)$?
- Depends on curvature, given by second derivative
- Hessian: $H_\theta = \frac{\delta^2}{\delta \theta_i \delta \theta_j} J(\theta)$
- Using Taylor expansion, impact of update $\theta - \epsilon \nabla_\theta$,

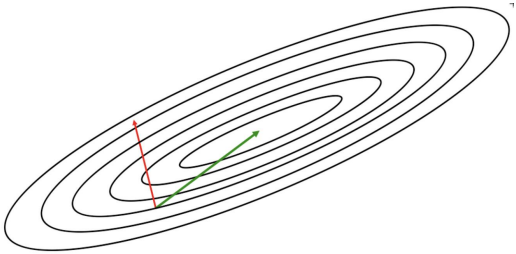
$$J(\theta) - \nabla_\theta^T \nabla_\theta + \frac{1}{2} \nabla_\theta^T H_\theta \nabla_\theta$$

- Analyze H_θ to check for ill conditioning



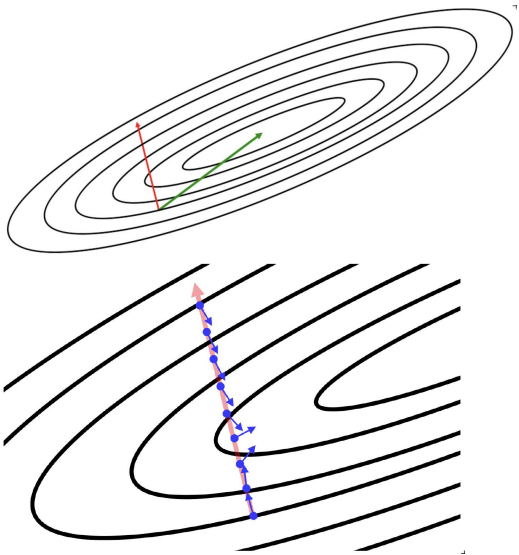
Directing gradient descent

- Locally steepest direction of descent may be far from the optimum
  - Elliptical contours vs circular contours



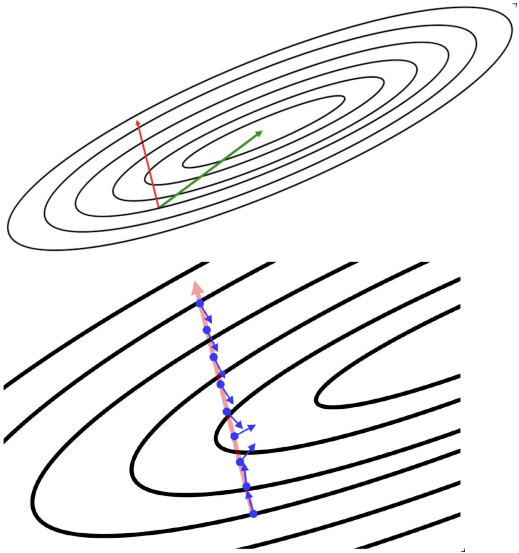
Directing gradient descent

- Locally steepest direction of descent may be far from the optimum
  - Elliptical contours vs circular contours
- Gradient changes rapidly along the direction of steepest descent
  - Taking large steps is problematic



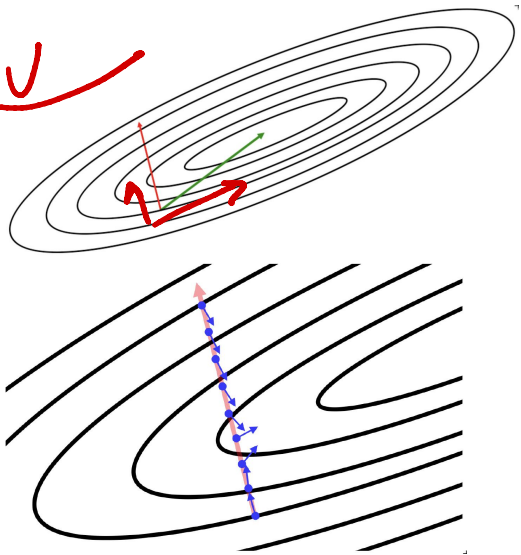
Directing gradient descent

- Locally steepest direction of descent may be far from the optimum
  - Elliptical contours vs circular contours
- Gradient changes rapidly along the direction of steepest descent
  - Taking large steps is problematic
- Ill-conditioned Hessian H — second derivatives
  - Computing Hessian is expensive
  - "Second order" methods are not used in practice



Directing gradient descent

- Locally steepest direction of descent may be far from the optimum
  - Elliptical contours vs circular contours
- Gradient changes rapidly along the direction of steepest descent
  - Taking large steps is problematic
- Ill-conditioned Hessian H — second derivatives
  - Computing Hessian is expensive
  - "Second order" methods are not used in practice
- Instead, heuristics like momentum and adaptive learning rates



Momentum

- SGD convergence can be very slow

Momentum

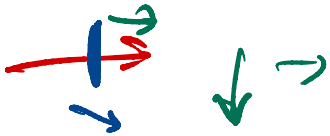
- SGD convergence can be very slow
- Momentum in physics — mass $\times$ velocity

Momentum

- SGD convergence can be very slow
- Momentum in physics — mass $\times$ velocity
- Introduce velocity v in SGD — assume unit mass
  - Moving average of past gradients, exponential decay
  - If gradient remains steady, velocity increases

Momentum

- SGD convergence can be very slow
- Momentum in physics — mass $\times$ velocity
- Introduce velocity v in SGD — assume unit mass
  - Moving average of past gradients, exponential decay
  - If gradient remains steady, velocity increases
- Update rule
  - $v \leftarrow \alpha v - \epsilon \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^{m} L(f(x_i; \theta), y_i) \right)$
  - $\theta \leftarrow \theta + v$



$J(\theta)$

$\theta \leftarrow \theta - \epsilon \nabla_\theta J$

$\alpha < 1$

Momentum

- SGD convergence can be very slow
- Momentum in physics — mass $\times$ velocity
- Introduce velocity v in SGD — assume unit mass
  - Moving average of past gradients, exponential decay
  - If gradient remains steady, velocity increases
- Update rule
  - $v \leftarrow \alpha v - \epsilon \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^{m} L(f(x_i; \theta), y_i) \right)$
  - $\theta \leftarrow \theta + v$
- Hyperparameter $\alpha \in [0, 1)$ — "friction", exponentially decaying history
  - With constant gradient g, in the limit $\frac{\epsilon g}{1 - \alpha}$, geometric progression

$\alpha = 0.9$

$\frac{1}{1-\alpha} = 10$

Nesterov momentum optimization

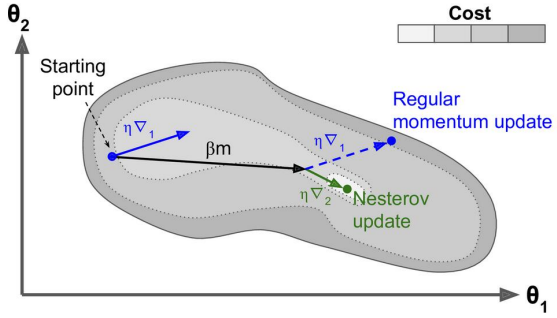
- Measure cost function slightly ahead, in direction of momentum

Nesterov momentum optimization

- Measure cost function slightly ahead, in direction of momentum
- Update rule
  - $v \leftarrow \alpha v - \epsilon \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^{m} L(f(x_i; \theta + \beta m), y_i) \right)$
  - $\theta \leftarrow \theta + v$

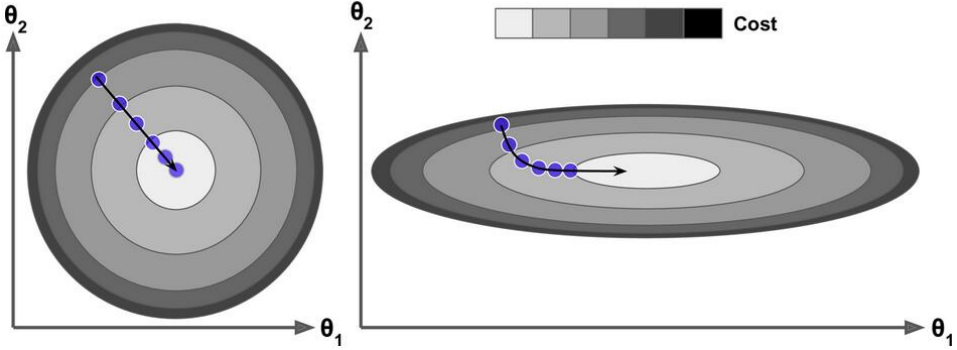
Nesterov momentum optimization

- Measure cost function slightly ahead, in direction of momentum
- Update rule
  - $v \leftarrow \alpha v - \epsilon \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^{m} L(f(x_i; \theta + \beta m), y_i) \right)$
  - $\theta \leftarrow \theta + v$
- Controls sideways oscillations better



Adjusting the trajectory

- If features have different scales, gradient descent is steeper in some dimensions
- How can we correct for this?



Adagrad

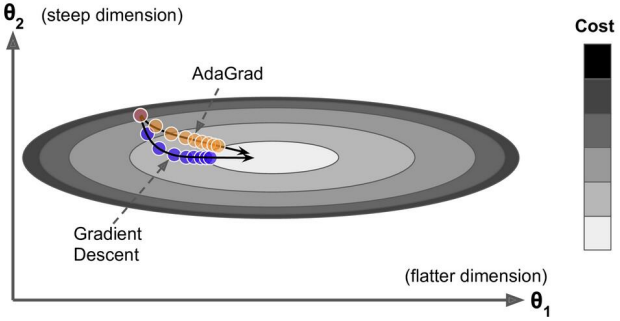
- Adagrad update rule
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $r \leftarrow r + g \cdot g$
  - $\Delta\theta \leftarrow \frac{\epsilon}{\delta + \sqrt{r}} \cdot g$, where $\delta \approx 10^{-7}$, for numerical stability
  - $\theta \leftarrow \theta + \Delta\theta$

Adagrad

- Adagrad update rule
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $r \leftarrow r + g \cdot g$
  - $\Delta\theta \leftarrow \frac{\epsilon}{\delta + \sqrt{r}} \cdot g$, where $\delta \approx 10^{-7}$, for numerical stability
  - $\theta \leftarrow \theta + \Delta\theta$
- Shrink learning rate in each dimension according to entire history of squared gradient

Adagrad

- Adagrad update rule
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $r \leftarrow r + g \cdot g$
  - $\Delta\theta \leftarrow \frac{\epsilon}{\delta + \sqrt{r}} \cdot g$, where $\delta \approx 10^{-7}$, for numerical stability
  - $\theta \leftarrow \theta + \Delta\theta$
- Shrink learning rate in each dimention according to entire history of squared gradient



Adaptive learning rates

RMSProp

- Using entire history shrinks learning rate too much
- Exponentially decaying average, discard extreme past
- Update rule
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $r \leftarrow \rho r + (1 - \rho)(g \cdot g)$, where ρ is decay rate
  - $\Delta\theta \leftarrow \frac{\epsilon}{\sqrt{\delta + r}} \cdot g$, where $\delta \approx 10^{-6}$
  - $\theta \leftarrow \theta + \Delta\theta$
- New hyperparameter ρ

Adam

- Adaptive moments — combines RMSProp and moments

Adam

- Adaptive moments — combines RMSProp and moments
- Update rule
  - Step size ϵ; two decay rates ρ_1, ρ_2; two moments, $s = r = 0$; time step $t = 0$
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $s \leftarrow \rho_1 s + (1 - \rho_1) g$; $r \leftarrow \rho_2 r + (1 - \rho_2)(g \cdot g)$
  - Correct bias in first and second moments: $\hat{s} \leftarrow \frac{s}{1 - \rho_1^t}$, $\hat{r} \leftarrow \frac{r}{1 - \rho_2^t}$
  - $\Delta\theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$; $\theta \leftarrow \theta + \Delta\theta$

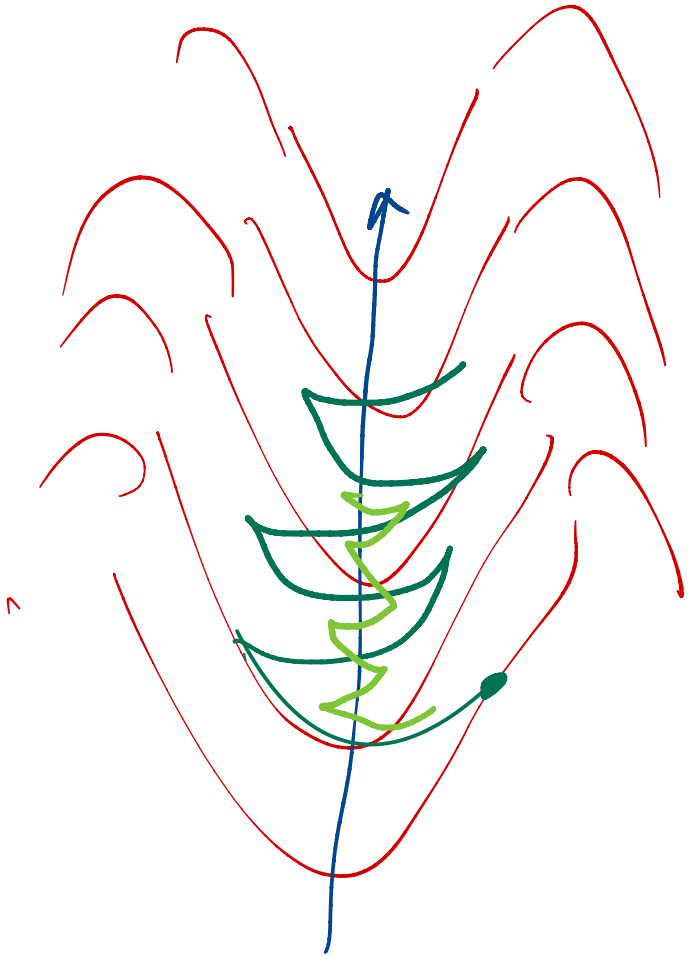
velocity

RMS Prop

$V \leftarrow \alpha V - \epsilon \nabla_\theta J$

$\theta := \theta + V$

Variants on Adam

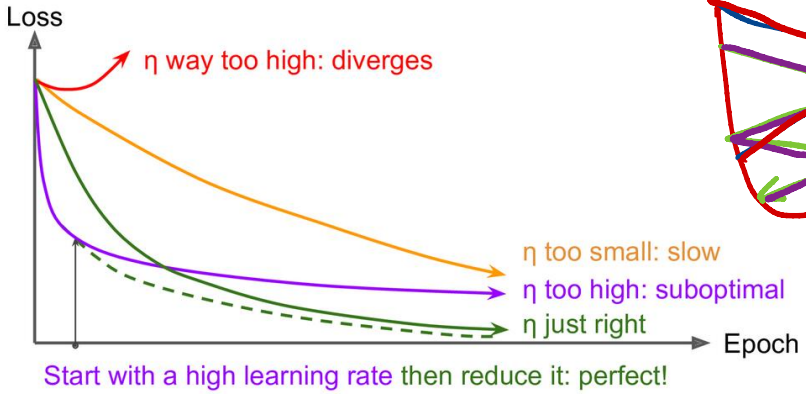


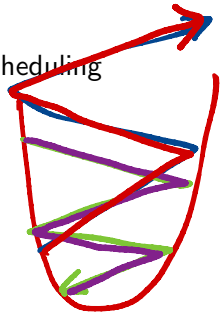
Adam

- Adaptive moments — combines RMSProp and moments
- Update rule
  - Step size ϵ; two decay rates ρ_1, ρ_2; two moments, $s = r = 0$; time step $t = 0$
  - $g \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} L(f(x_i; \theta), y_i)$
  - $s \leftarrow \rho_1 s + (1 - \rho_1) g$; $r \leftarrow \rho_2 r + (1 - \rho_2)(g \cdot g)$
  - Correct bias in first and second moments: $\hat{s} \leftarrow \frac{s}{1 - \rho_1^t}$, $\hat{r} \leftarrow \frac{r}{1 - \rho_2^t}$
  - $\Delta\theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$; $\theta \leftarrow \theta + \Delta\theta$
- No clear theoretical justification for combinining momentum and scaling
- Fairly robust with respect to values of hyperparameters

Adaptive learning rates

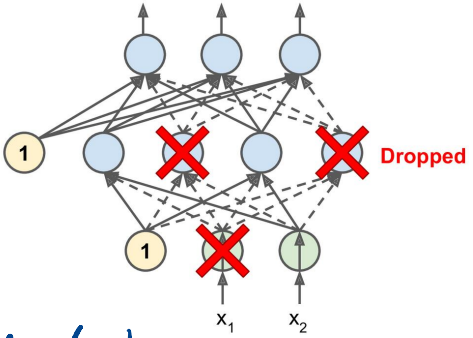
- Choosing a fixed learning rate is hard
- Make a learning rate a function of iteration number
- Power scheduling, exponential scheduling, piecewise constant scheduling





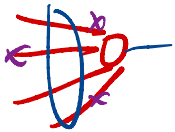
Regularization

- ℓ_1 and ℓ_2 regularization, as usual
- Dropout
  - Disable nodes with probability p
  - Analogy — multifunctional employees
  - Scale weights after training



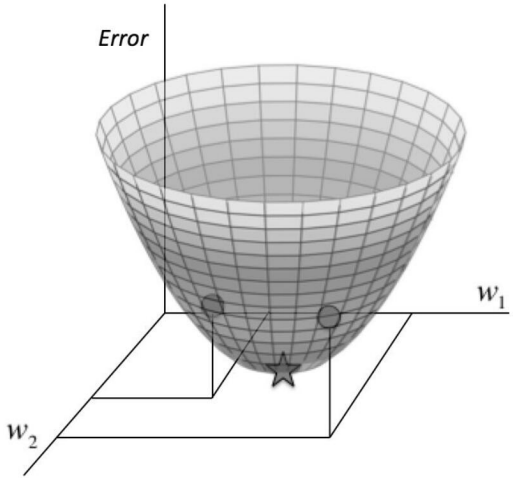
Original Cost + Parameter Cost

Regularization



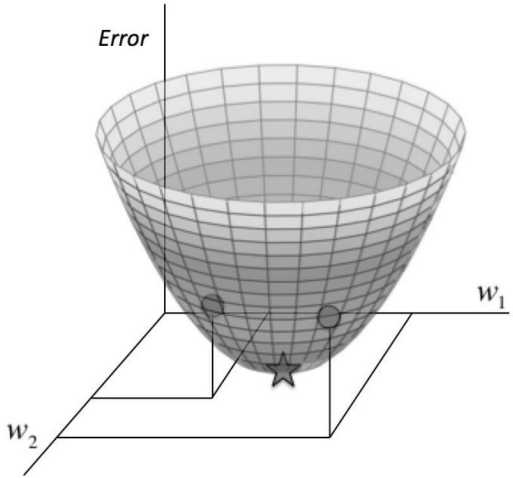
Local minima

- The loss function for regression is convex
- Gradient descent converges to global optimum



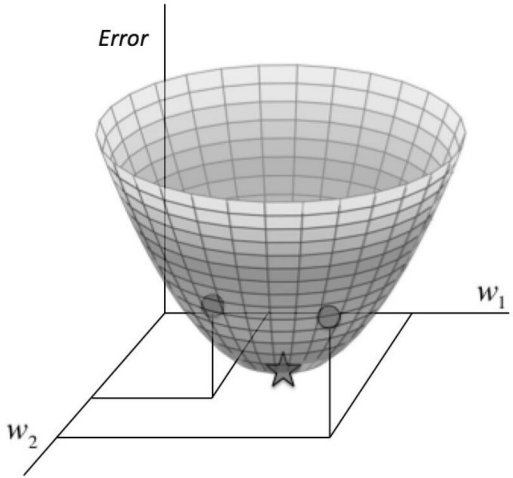
Local minima

- The loss function for regression is convex
- Gradient descent converges to global optimum
- Loss function for neural networks is not convex
- In general, gradient descent only finds local minima



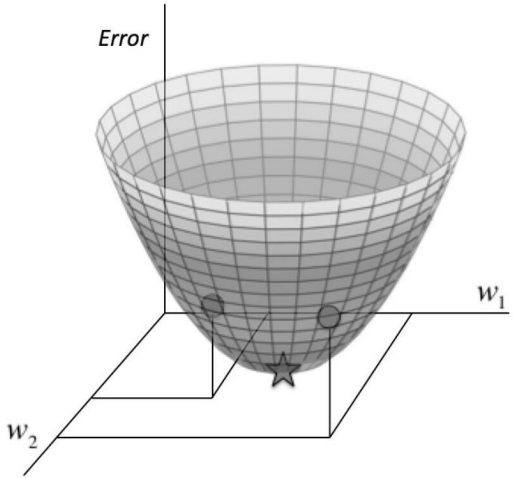
Local minima

- The loss function for regression is convex
- Gradient descent converges to global optimum
- Loss function for neural networks is not convex
- In general, gradient descent only finds local minima
- How many local minima are there?



Local minima

- The loss function for regression is convex
- Gradient descent converges to global optimum
- Loss function for neural networks is not convex
- In general, gradient descent only finds local minima
- How many local minima are there?
- How does it affect gradient descent?



Model identifiability

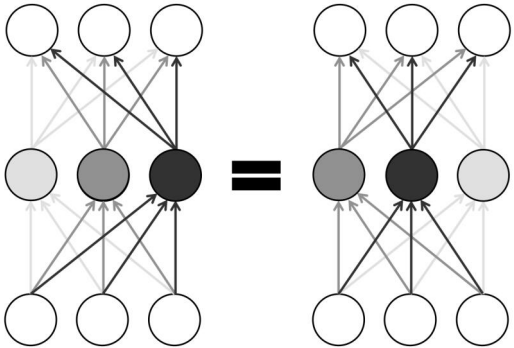
- Is the model that fits the data unique?

Model identifiability

- Is the model that fits the data unique?
- Non-identifiable — two or more settings of the parameters are observationally equivalent

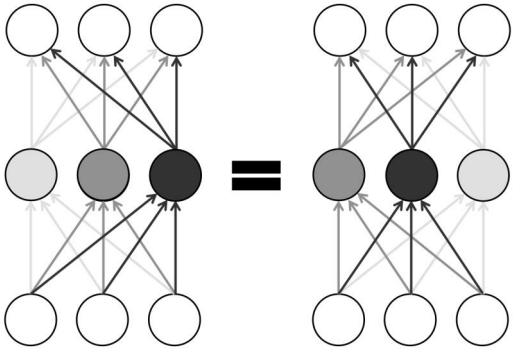
Model identifiability

- Is the model that fits the data unique?
- Non-identifiable — two or more settings of the parameters are observationally equivalent
- Symmetry
  - Fully connected network, permutations of a layer are indistinguishable



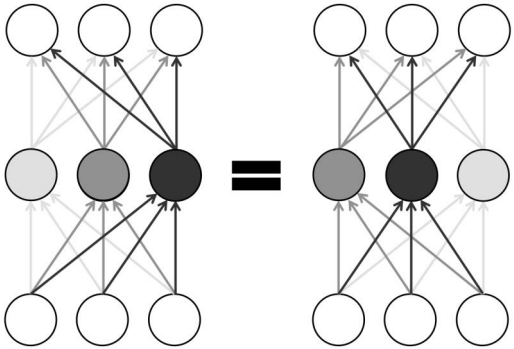
Model identifiability

- Is the model that fits the data unique?
- Non-identifiable — two or more settings of the parameters are observationally equivalent
- Symmetry
  - Fully connected network, permutations of a layer are indistinguishable
- Piecewise linear activation — ReLU
  - Scale inputs by k, multiply output by $1/k$



Model identifiability

- Is the model that fits the data unique?
- Non-identifiable — two or more settings of the parameters are observationally equivalent
- Symmetry
  - Fully connected network, permutations of a layer are indistinguishable
- Piecewise linear activation — ReLU
  - Scale inputs by k, multiply output by $1/k$
- Large numbers of local minima!



How problematic are local minima?

- How to measure the impact of local minima?

How problematic are local minima?

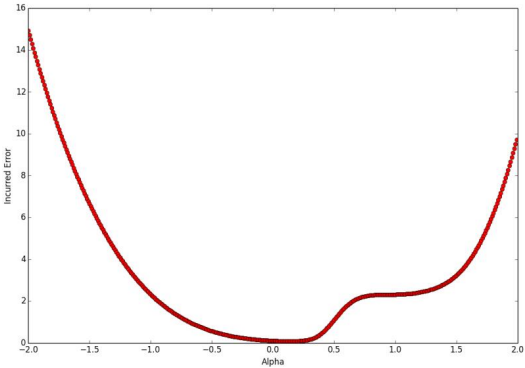
- How to measure the impact of local minima?
- Training process will see local ups and downs, but "bumpy" surface may not give a good picture

How problematic are local minima?

- How to measure the impact of local minima?
- Training process will see local ups and downs, but "bumpy" surface may not give a good picture
- Instead [Goodfellow et al]
  - Random initialization θ_i
  - SGD finds an optimum value θ_f

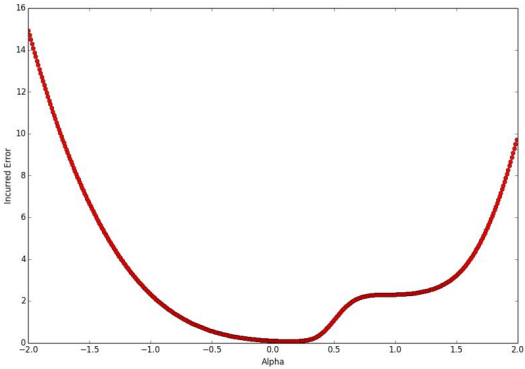
How problematic are local minima?

- How to measure the impact of local minima?
- Training process will see local ups and downs, but "bumpy" surface may not give a good picture
- Instead [Goodfellow et al]
  - Random initialization θ_i
  - SGD finds an optimum value θ_f
  - Check loss along the linearly interpolation $\theta_\alpha = \alpha \cdot \theta_f + (1 - \alpha) \cdot \theta_i$



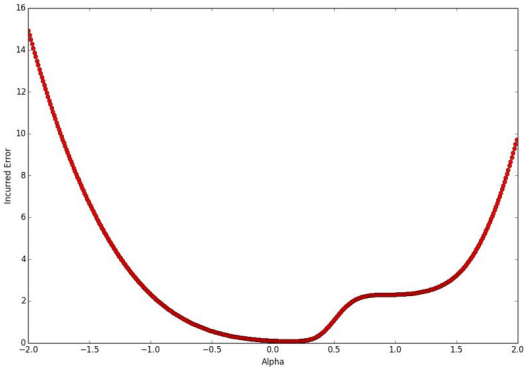
How problematic are local minima?

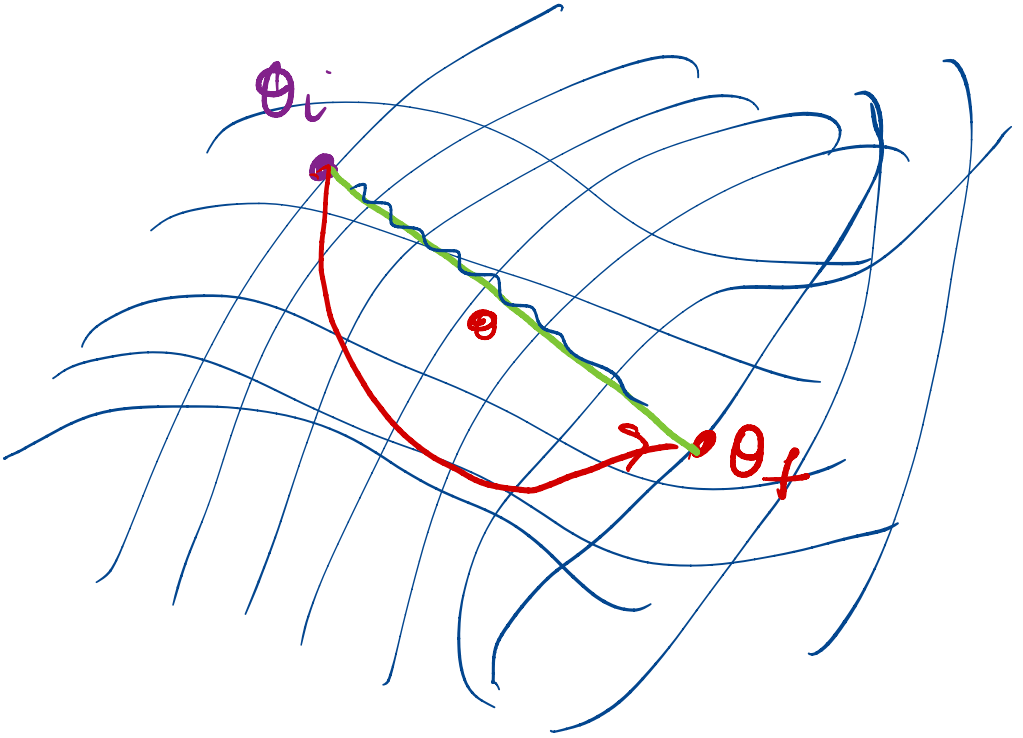
- How to measure the impact of local minima?
- Training process will see local ups and downs, but "bumpy" surface may not give a good picture
- Instead [Goodfellow et al]
  - Random initialization θ_i
  - SGD finds an optimum value θ_f
  - Check loss along the linearly interpolation $\theta_\alpha = \alpha \cdot \theta_f + (1 - \alpha) \cdot \theta_i$
  - Are there problematic local minima along the path?



How problematic are local minima?

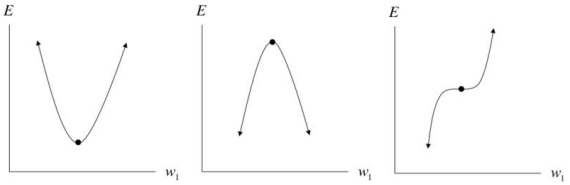
- How to measure the impact of local minima?
- Training process will see local ups and downs, but "bumpy" surface may not give a good picture
- Instead [Goodfellow et al]
  - Random initialization θ_i
  - SGD finds an optimum value θ_f
  - Check loss along the linearly interpolation $\theta_\alpha = \alpha \cdot \theta_f + (1 - \alpha) \cdot \theta_i$
  - Are there problematic local minima along the path?
- Typically, no!




θ_i
θ_f

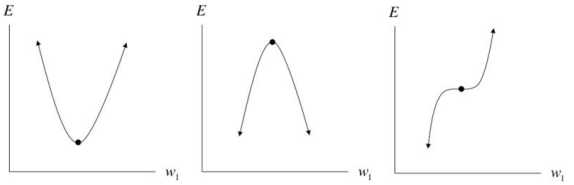
Saddle points

- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima



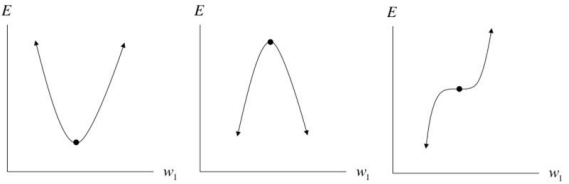
Saddle points

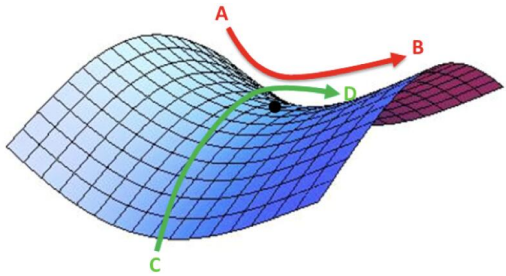
- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima



Saddle points

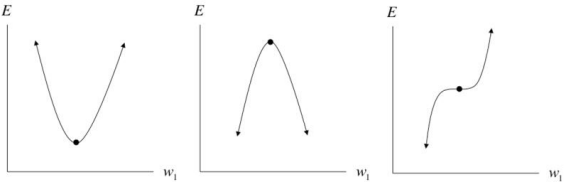
- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima
- Large fraction of critical values are saddle points

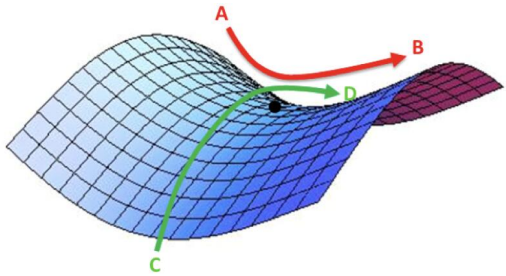




Saddle points

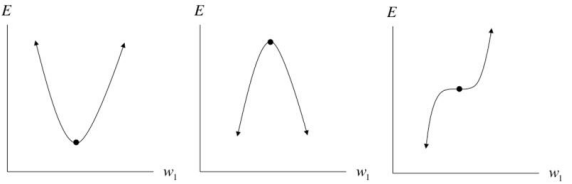
- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima
- Large fraction of critical values are saddle points
- Does not seem to be a problem for SGD

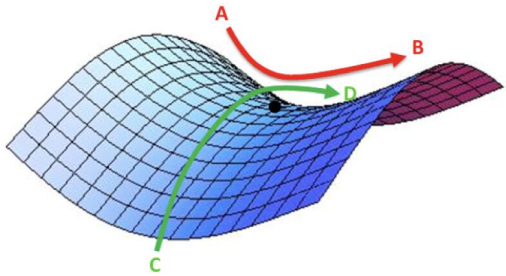




Saddle points

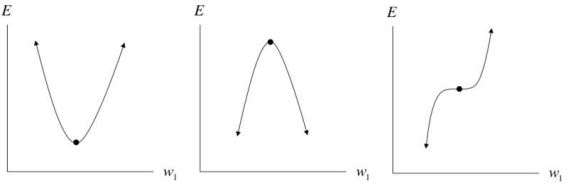
- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima
- Large fraction of critical values are saddle points
- Does not seem to be a problem for SGD
- Solving directly for zero gradient is problematic

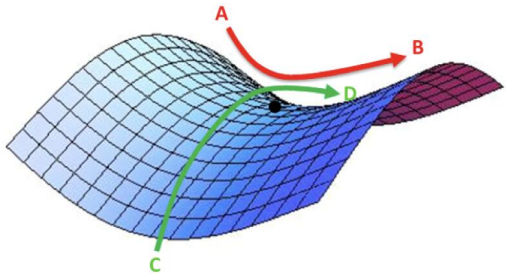




Saddle points

- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima
- Large fraction of critical values are saddle points
- Does not seem to be a problem for SGD
- Solving directly for zero gradient is problematic





Saddle points

- Critical points — zero gradient
  - Minimum, maximum or inflection point
  - k critical points $\rightarrow k/3$ are minima
- In d dimensions
  - Should be minimum in d directions
  - k critical points $\rightarrow k/3^d$ are minima
- Large fraction of critical values are saddle points
- Does not seem to be a problem for SGD
- Solving directly for zero gradient is problematic