Recap: Optimisation

L_2 Regularization

$$\hat{L}_R(\boldsymbol{\theta}) = \hat{L}(\boldsymbol{\theta}) + \frac{\beta}{2}\|\boldsymbol{\theta}\|^2$$

- Effect on gradient descent update

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha^{(t)}\,\nabla_{\boldsymbol{\theta}}\,\hat{L}\left(\boldsymbol{\theta}^{(t)}\right) - \alpha^{(t)}\beta\boldsymbol{\theta}^{(t)}$$

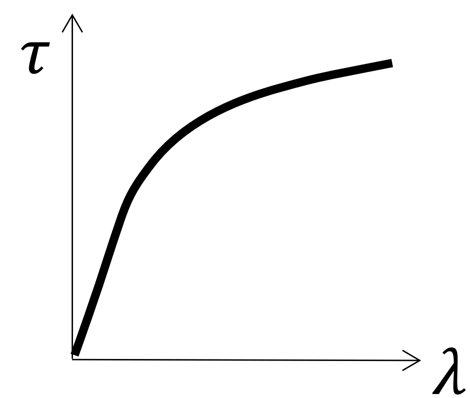
$$= \left(1 - \alpha^{(t)}\beta\right)\boldsymbol{\theta}^{(t)} - \alpha^{(t)}\,\nabla_{\boldsymbol{\theta}}\,\hat{L}\left(\boldsymbol{\theta}^{(t)}\right)$$

"weight decay"

- Bayesian interpretation: $\boldsymbol{\theta} \sim \mathcal{N}\left(0, \frac{1}{\sqrt{2\beta}}\mathbf{I}\right)$

- **Filter of $\boldsymbol{\theta}^*$** (attenuates along small eigenvectors)

$$\begin{aligned}\boldsymbol{\theta}_R^* &\approx \left(\nabla_{\boldsymbol{\theta}}^2\hat{L}(\boldsymbol{\theta}^*) + \beta\mathbf{I}\right)^{-1}\nabla_{\boldsymbol{\theta}}^2\hat{L}(\boldsymbol{\theta}^*)\boldsymbol{\theta}^* \\ &\approx \mathbf{U}(\boldsymbol{\Lambda} + \beta\mathbf{I})^{-1}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}\boldsymbol{\theta}^* \\ &= \mathbf{U}\tau(\boldsymbol{\Lambda})\mathbf{U}^{\mathrm{T}}\boldsymbol{\theta}^* \\ &= \tau\left(\nabla_{\boldsymbol{\theta}}^2\hat{L}(\boldsymbol{\theta}^*)\right)\boldsymbol{\theta}^*\end{aligned}$$



Noisy input

- Add i.i.d. noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \beta \mathbf{I})$ to input
- Loss becomes

$$
\begin{aligned}
L(\mathbf{w}) &= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}(\mathbf{x}+\boldsymbol{\epsilon})-y\right)^2 \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}-y\right)^2 + 2\mathbb{E}\,\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\epsilon}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}-y\right) + \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\epsilon}\right)^2 \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}-y\right)^2 + \mathbf{w}^{\mathrm{T}}\mathbb{E}\left(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}}\right)\mathbf{w} \\
&= \mathbb{E}\left(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}-y\right)^2 + \beta\|\mathbf{w}\|^2
\end{aligned}
$$

- Equivalent to weight decay!

Descent method: general recipe

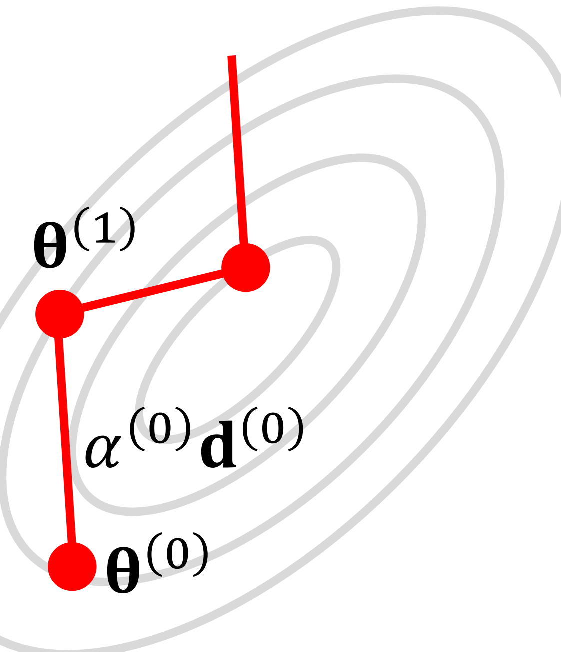
Initialization: start with some $\boldsymbol{\theta}^{(0)}$

For $k = 0, \ldots$ **until convergence**

Choose **descent direction** $\mathbf{d}^{(k)} = -\nabla \hat{L}\left(\boldsymbol{\theta}^{(k)}\right)$

Choose **step size** $\alpha^{(k)}$

Update $\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} + \alpha^{(k)} \mathbf{d}^{(k)}$



Gradient descent convergence rate

Strong convexity $\nabla^2 \hat{L}(\boldsymbol{\theta}) \succcurlyeq m\mathbf{I} \quad m > 0$

Lipschitz gradient $\nabla^2 \hat{L}(\boldsymbol{\theta}) \preccurlyeq M\mathbf{I}$

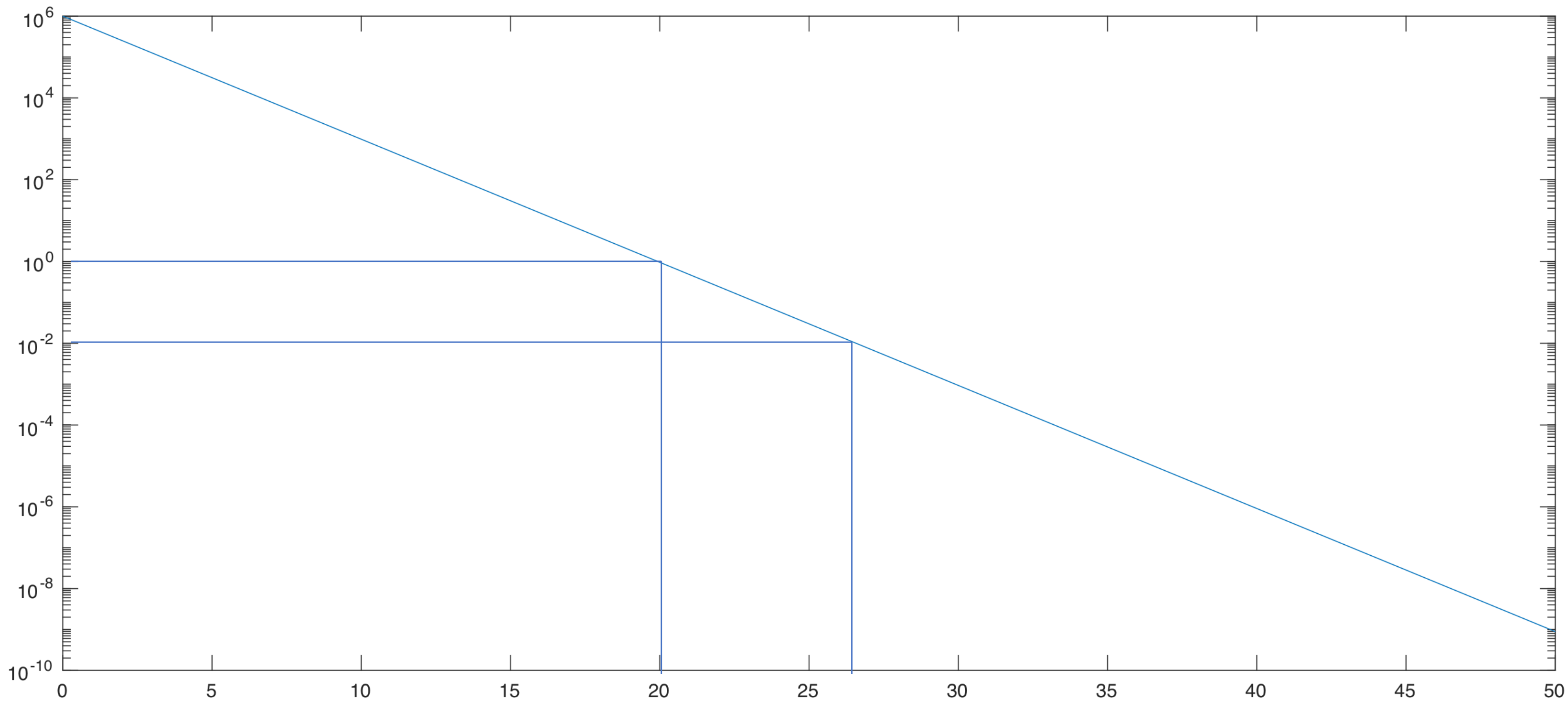
Constant step size $\alpha \leq \dfrac{2}{m+M}$

$$\hat{L}\left(\boldsymbol{\theta}^{(k)}\right) - \hat{L}(\boldsymbol{\theta}^*) \leq c^k \cdot \tfrac{M}{2}\left\|\boldsymbol{\theta}^{(0)} - \boldsymbol{\theta}^{(k)}\right\|$$

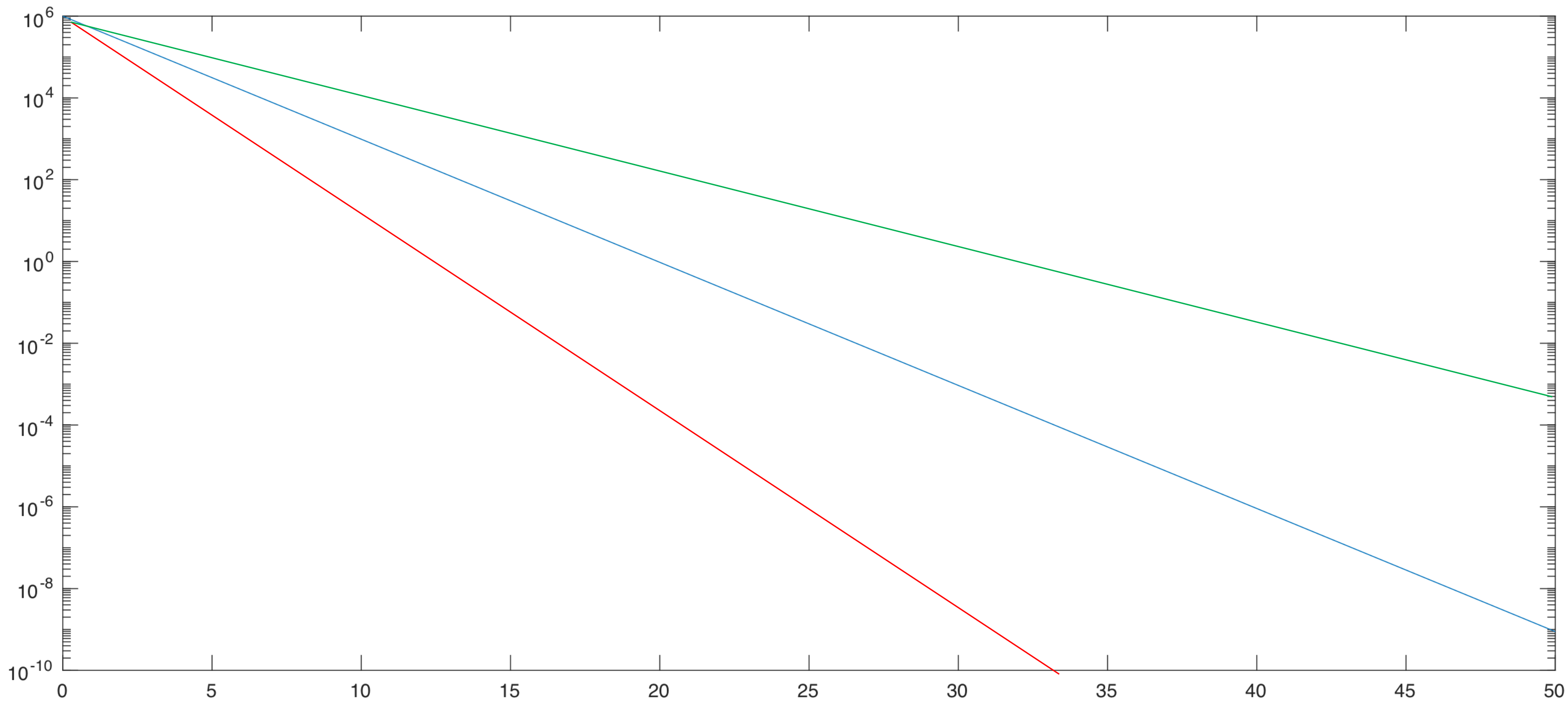
"Linear" convergence

To get $\hat{L}\left(\boldsymbol{\theta}^{(k)}\right) - \hat{L}(\boldsymbol{\theta}^*) \leq \epsilon$ one needs $\mathcal{O}\left(\log\dfrac{1}{\epsilon}\right)$ iterations

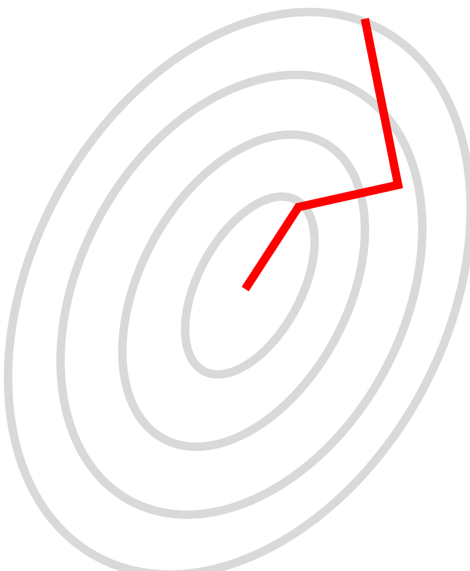
Gradient descent convergence rate



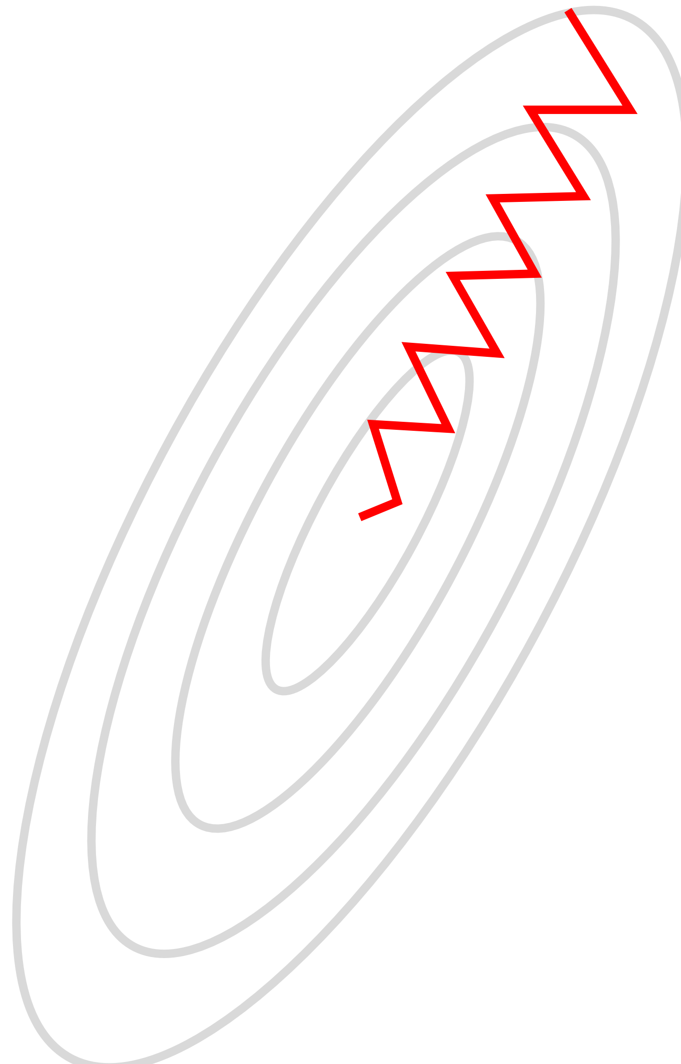
Gradient descent convergence rate



Gradient descent convergence rate



$$c = 1 - \frac{m}{M}$$



Momentum

- Key idea: add back fraction of past step

$$\Delta\boldsymbol{\theta}^{(k)} = -\alpha\nabla\hat{L}\left(\boldsymbol{\theta}^{(k)}\right) + \beta\Delta\boldsymbol{\theta}^{(k-1)} \qquad 0 < \beta < 1$$

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \Delta\boldsymbol{\theta}^{(k)}$$

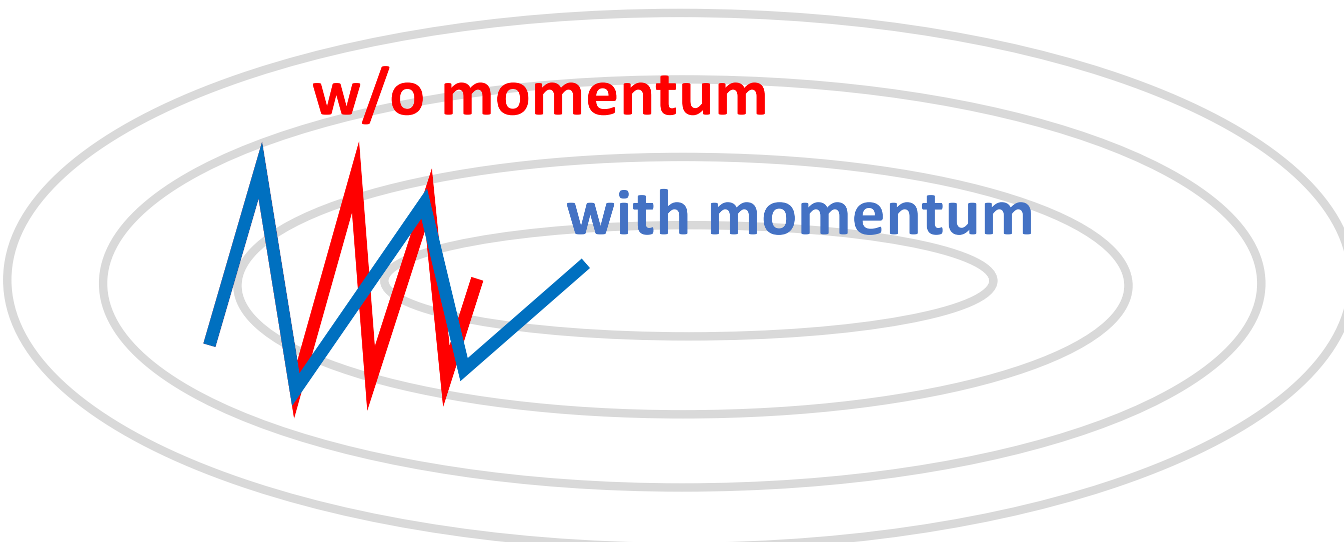
- Exponentially decaying memory

$$\Delta\boldsymbol{\theta}^{(k)} = -\alpha\left[\nabla\hat{L}\left(\boldsymbol{\theta}^{(k)}\right) + \beta\nabla\hat{L}\left(\boldsymbol{\theta}^{(k-1)}\right) + \cdots \beta^{n}\nabla\hat{L}\left(\boldsymbol{\theta}^{(k-n)}\right)\right] + \beta^{n+1}\Delta\boldsymbol{\theta}^{(k-n-1)}$$

- Dampen oscillations of gradient descent

Rumelhart et al. 1986

Momentum



AdaGrad

- Key idea: use different update per coordinate

$$\theta_i^{(k+1)} = \theta_i^{(k)} - \frac{\alpha}{\sqrt{g_{ii}^{(k)} + \epsilon}} \frac{\partial}{\partial \theta_i} \hat{L}\left(\boldsymbol{\theta}^{(k)}\right)$$

$$\mathbf{G}^{(k)} = \sum_{l=1}^{k} \nabla \hat{L}\left(\boldsymbol{\theta}^{(l)}\right) \nabla \hat{L}\left(\boldsymbol{\theta}^{(l)}\right)^{\mathrm{T}}$$

Covariance up to scaling factor

Duchi et al. 2011

AdaGrad

- Key idea: use different update per coordinate

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha\left(\mathrm{Diag}\mathbf{G}^{(k)} + \epsilon\mathbf{I}\right)^{-1/2}\nabla\hat{L}\left(\boldsymbol{\theta}^{(k)}\right)$$

Matrix without off-diag elements

$$\mathbf{G}^{(k)} = \sum_{l=1}^{k}\nabla\hat{L}\left(\boldsymbol{\theta}^{(l)}\right)\nabla\hat{L}\left(\boldsymbol{\theta}^{(l)}\right)^{\mathrm{T}}$$

- Eliminates the need to manually tune learning rate

- Accumulated sum in denominator makes learning rate too small

Duchi et al. 2011

AdaDelta or RMSProp

- Modification of AdaGrad using decaying average of gradients

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \frac{\alpha}{\sqrt{\mathbf{g}^{(k)} + \epsilon}} \nabla \hat{L}\left(\boldsymbol{\theta}^{(k)}\right)$$

$$\mathbf{g}^{(k)} = \gamma \mathbf{g}^{(k-1)} + (1-\gamma) \nabla \hat{L}^2\left(\boldsymbol{\theta}^{(k)}\right)$$

* All operations performed coordinate-wise
Zeiler 2012; Hinton (unpublished)

Adam (Adaptive Moment Estimation)

- Key idea: keep decaying average of gradients and their squares

$$\mathbf{m}^{(k)} = \beta_1 \mathbf{m}^{(k-1)} + (1-\beta_1)\nabla \hat{L}\left(\boldsymbol{\theta}^{(k)}\right)$$
$$\mathbf{v}^{(k)} = \beta_2 \mathbf{v}^{(k-1)} + (1-\beta_2)\nabla \hat{L}^2\left(\boldsymbol{\theta}^{(k)}\right)$$

$$\widehat{\mathbf{m}}^{(k)} = \frac{\mathbf{m}^{(k)}}{1-\beta_1^k} \qquad \hat{\mathbf{v}}^{(k)} = \frac{\mathbf{v}^{(k)}}{1-\beta_2^k}$$

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha \frac{\widehat{\mathbf{m}}^{(k)}}{\sqrt{\hat{\mathbf{v}}^{(k)}} + \epsilon}$$

Kingma et al. 2015