by repeatedly applying the same operation at each time step of a long temporal sequence. Repeated application of the same parameters gives rise to especially pronounced difficulties.

For example, suppose that a computational graph contains a path that consists of repeatedly multiplying by a matrix $\boldsymbol{W}$. After t steps, this is equivalent to multiplying by $\boldsymbol{W}^t$. Suppose that $\boldsymbol{W}$ has an eigendecomposition $\boldsymbol{W} = \boldsymbol{V}\text{diag}(\boldsymbol{\lambda})\boldsymbol{V}^{-1}$. In this simple case, it is straightforward to see that

$$\boldsymbol{W}^t = \left(\boldsymbol{V}\text{diag}(\boldsymbol{\lambda})\boldsymbol{V}^{-1}\right)^t = \boldsymbol{V}\text{diag}(\boldsymbol{\lambda})^t\boldsymbol{V}^{-1}. \tag{8.11}$$

Any eigenvalues λ_i that are not near an absolute value of 1 will either explode if they are greater than 1 in magnitude or vanish if they are less than 1 in magnitude. The **vanishing and exploding gradient problem** refers to the fact that gradients through such a graph are also scaled according to $\text{diag}(\boldsymbol{\lambda})^t$. Vanishing gradients make it difficult to know which direction the parameters should move to improve the cost function, while exploding gradients can make learning unstable. The cliff structures described earlier that motivate gradient clipping are an example of the exploding gradient phenomenon.

The repeated multiplication by $\boldsymbol{W}$ at each time step described here is very similar to the **power method** algorithm used to find the largest eigenvalue of a matrix $\boldsymbol{W}$ and the corresponding eigenvector. From this point of view it is not surprising that $\boldsymbol{x}^\top\boldsymbol{W}^t$ will eventually discard all components of $\boldsymbol{x}$ that are orthogonal to the principal eigenvector of $\boldsymbol{W}$.

Recurrent networks use the same matrix $\boldsymbol{W}$ at each time step, but feedforward networks do not, so even very deep feedforward networks can largely avoid the vanishing and exploding gradient problem (Sussillo, 2014).

We defer a further discussion of the challenges of training recurrent networks until section 10.7, after recurrent networks have been described in more detail.

8.2.6 Inexact Gradients

Most optimization algorithms are designed with the assumption that we have access to the exact gradient or Hessian matrix. In practice, we usually only have a noisy or even biased estimate of these quantities. Nearly every deep learning algorithm relies on sampling-based estimates at least insofar as using a minibatch of training examples to compute the gradient.

In other cases, the objective function we want to minimize is actually intractable. When the objective function is intractable, typically its gradient is intractable as well. In such cases we can only approximate the gradient. These issues mostly arise