

# Автоматическое дифференцирование.

Семинар

Оптимизация для всех! ЦУ



## Автоматическое дифференцирование

## Прямой режим



Рис. 1: Иллюстрация прямого режима для вычисления производной функции  $v_i$  по отношению к  $w_k$ .

- Использует прямой chain rule

## Прямой режим



Рис. 1: Иллюстрация прямого режима для вычисления производной функции  $v_i$  по отношению к  $w_k$ .

- Использует прямой chain rule
- Имеет сложность  $d \times \mathcal{O}(T)$  операций

## Обратный режим



Рис. 2: Иллюстрация обратного режима для вычисления производной функции  $L$  по отношению к узлу  $v_i$ .

- Использует обратный chain rule

## Обратный режим



Рис. 2: Иллюстрация обратного режима для вычисления производной функции  $L$  по отношению к узлу  $v_i$ .

- Использует обратный chain rule
- Хранит информацию из прямого прохода

## Обратный режим



Рис. 2: Иллюстрация обратного режима для вычисления производной функции  $L$  по отношению к узлу  $v_i$ .

- Использует обратный chain rule
- Хранит информацию из прямого прохода
- Имеет сложность  $\mathcal{O}(T)$  операций

## Задачи по автоматическому дифференцированию

## Простой пример

### Example

$$f(x_1, x_2) = x_1 * x_2 + \sin x_1$$

Давайте вычислим производные  $\frac{\partial f}{\partial x_i}$  используя прямой и обратный режимы.

## Простой пример

### Example

$$f(x_1, x_2) = x_1 * x_2 + \sin x_1$$

Давайте вычислим производные  $\frac{\partial f}{\partial x_i}$  используя прямой и обратный режимы.

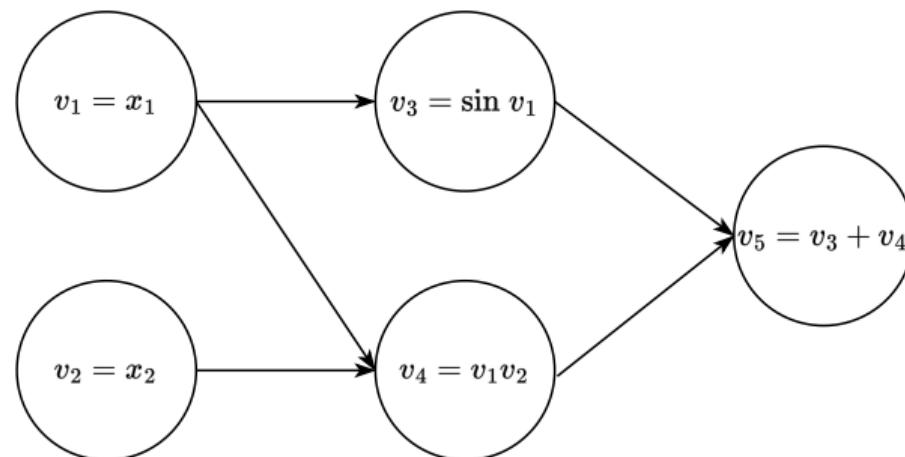


Рис. 3: Иллюстрация вычислительного графа функции  $f(x_1, x_2)$ .

# Автоматическое дифференцирование с JAX

## Пример 1

$$f(X) = \text{tr}(AX^{-1}B)$$

$$\nabla f = -X^{-T}A^TB^TX^{-T}$$

# Автоматическое дифференцирование с JAX

## Пример 1

$$f(X) = \text{tr}(AX^{-1}B)$$

$$\nabla f = -X^{-T}A^TB^TX^{-T}$$

## Пример 2

$$g(x) = 1/3 \cdot \|x\|_2^3$$

$$\nabla^2 g = \|x\|_2^{-1}xx^T + \|x\|_2 I_n$$

# Автоматическое дифференцирование с JAX

## Пример 1

$$f(X) = \text{tr}(AX^{-1}B)$$

$$\nabla f = -X^{-T}A^TB^TX^{-T}$$

## Пример 2

$$g(x) = 1/3 \cdot \|x\|_2^3$$

$$\nabla^2 g = \|x\|_2^{-1}xx^T + \|x\|_2 I_n$$

Давайте вычислим градиенты и гессианы  $f$  и  $g$  в python 

## Задача 1

### Question

Какой из режимов AD вы бы выбрали (прямой/обратный) для следующего вычислительного графа арифметических операций?

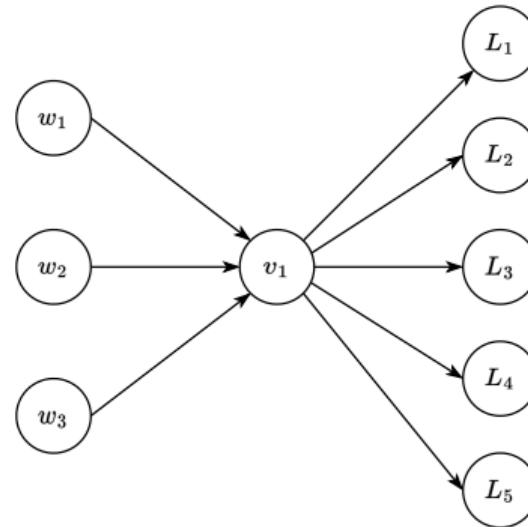


Рис. 4: Какой режим вы бы выбрали для вычисления градиентов?

## Задача 2

Предположим, у нас есть обратимая матрица  $A$  и вектор  $b$ , вектор  $x$  является решением системы линейных уравнений  $Ax = b$ , то есть можно записать аналитическое решение  $x = A^{-1}b$ .

### Question

Найдите производные  $\frac{\partial L}{\partial A}$ ,  $\frac{\partial L}{\partial b}$ .

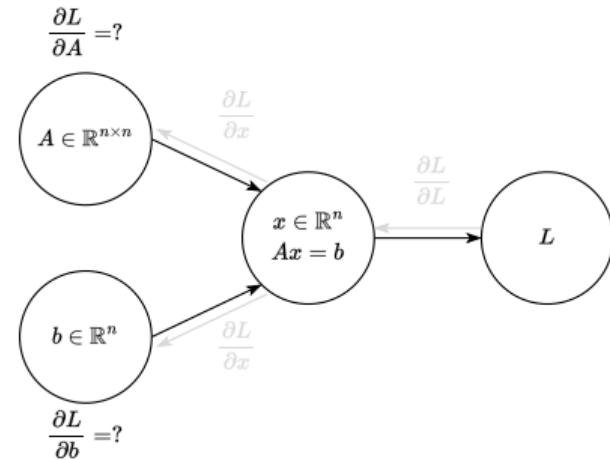


Рис. 5:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



Предположим, у нас есть обратимая матрица  $A$  и вектор  $b$ , вектор  $x$  является решением системы линейных уравнений  $Ax = b$ , то есть можно записать аналитическое решение  $x = A^{-1}b$ , в этом примере показано, что вычисление всех производных  $\frac{\partial L}{\partial A}, \frac{\partial L}{\partial b}, \frac{\partial L}{\partial x}$ , то есть обратный проход, стоит приблизительно столько же, сколько и прямой проход.

Рис. 6:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



Предположим, у нас есть обратимая матрица  $A$  и вектор  $b$ , вектор  $x$  является решением системы линейных уравнений  $Ax = b$ , то есть можно записать аналитическое решение  $x = A^{-1}b$ , в этом примере показано, что вычисление всех производных  $\frac{\partial L}{\partial A}, \frac{\partial L}{\partial b}, \frac{\partial L}{\partial x}$ , то есть обратный проход, стоит приблизительно столько же, сколько и прямой проход.

Известно, что дифференциал функции не зависит от параметризации:

$$dL = \left\langle \frac{\partial L}{\partial x}, dx \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

Рис. 6:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



Предположим, у нас есть обратимая матрица  $A$  и вектор  $b$ , вектор  $x$  является решением системы линейных уравнений  $Ax = b$ , то есть можно записать аналитическое решение  $x = A^{-1}b$ , в этом примере показано, что вычисление всех производных  $\frac{\partial L}{\partial A}, \frac{\partial L}{\partial b}, \frac{\partial L}{\partial x}$ , то есть обратный проход, стоит приблизительно столько же, сколько и прямой проход.

Известно, что дифференциал функции не зависит от параметризации:

$$dL = \left\langle \frac{\partial L}{\partial x}, dx \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

Для линейной системы мы имеем:

$$Ax = b$$

$$dAx + Adx = db \rightarrow dx = A^{-1}(db - dAx)$$

Рис. 6:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



Прямая подстановка дает нам:

$$\left\langle \frac{\partial L}{\partial x}, A^{-1}(db - dAx) \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

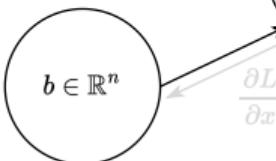
Рис. 7:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



$$Ax = b$$



$$\frac{\partial L}{\partial b} = ?$$

$$\frac{\partial L}{\partial b} = ?$$

Прямая подстановка дает нам:

$$\left\langle \frac{\partial L}{\partial x}, A^{-1}(db - dAx) \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

$$\left\langle -A^{-T} \frac{\partial L}{\partial x} x^T, dA \right\rangle + \left\langle A^{-T} \frac{\partial L}{\partial x}, db \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

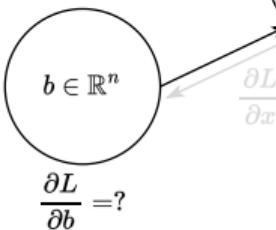
Рис. 7:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов

$$\frac{\partial L}{\partial A} = ?$$



$$Ax = b$$



$$\frac{\partial L}{\partial b} = ?$$

Прямая подстановка дает нам:

$$\left\langle \frac{\partial L}{\partial x}, A^{-1}(db - dAx) \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

$$\left\langle -A^{-T} \frac{\partial L}{\partial x} x^T, dA \right\rangle + \left\langle A^{-T} \frac{\partial L}{\partial x}, db \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

Следовательно:

$$\frac{\partial L}{\partial A} = -A^{-T} \frac{\partial L}{\partial x} x^T \quad \frac{\partial L}{\partial b} = A^{-T} \frac{\partial L}{\partial x}$$

Рис. 7:  $x$  может быть найден как решение линейной системы

## Распространение градиента через метод наименьших квадратов



Прямая подстановка дает нам:

$$\left\langle \frac{\partial L}{\partial x}, A^{-1}(db - dAx) \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

$$\left\langle -A^{-T} \frac{\partial L}{\partial x} x^T, dA \right\rangle + \left\langle A^{-T} \frac{\partial L}{\partial x}, db \right\rangle = \left\langle \frac{\partial L}{\partial A}, dA \right\rangle + \left\langle \frac{\partial L}{\partial b}, db \right\rangle$$

Следовательно:

$$\frac{\partial L}{\partial A} = -A^{-T} \frac{\partial L}{\partial x} x^T \quad \frac{\partial L}{\partial b} = A^{-T} \frac{\partial L}{\partial x}$$

Интересно, что наиболее вычислительно интенсивная часть здесь – это обратная матрица, которая является такой же, как и для прямого прохода. Иногда возможно хранить результат сам по себе, что делает обратный проход еще дешевле.

Рис. 7:  $x$  может быть найден как решение линейной системы

## Задача 3

Предположим, у нас есть прямоугольная матрица  $W \in \mathbb{R}^{m \times n}$ , которая имеет сингулярное разложение:

$$W = U\Sigma V^T, \quad U^T U = I, \quad V^T V = I,$$
$$\Sigma = \text{diag}(\sigma_1, \dots, \sigma_{\min(m,n)})$$

Регуляризатор  $R(W) = \text{tr}(\Sigma)$  в любой функции потерь стимулирует низкоранговые решения.

### Question

Найдите производную  $\frac{\partial R}{\partial W}$ .



Рис. 8: Вычислительный граф для сингулярного регуляризатора

# Распространение градиента через SVD



Предположим, у нас есть прямоугольная матрица  $W \in \mathbb{R}^{m \times n}$ , которая имеет сингулярное разложение:

$$W = U\Sigma V^T, \quad U^T U = I, \quad V^T V = I, \quad \Sigma = \text{diag}(\sigma_1, \dots, \sigma_{\min(m,n)})$$

1. Аналогично предыдущему примеру:

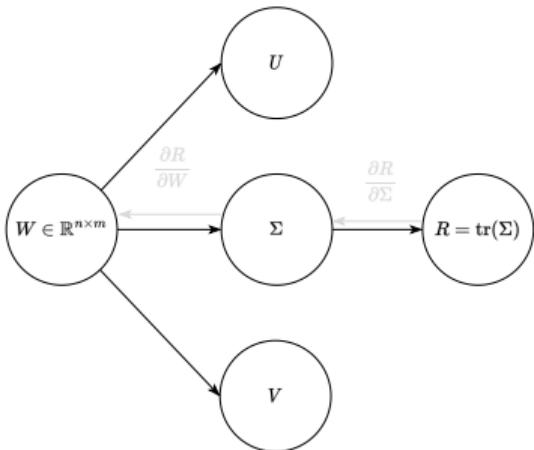
$$W = U\Sigma V^T$$

$$dW = dU\Sigma V^T + U d\Sigma V^T + U\Sigma dV^T$$

$$U^T dW V = U^T dU\Sigma V^T V + U^T U d\Sigma V^T V + U^T U\Sigma dV^T V$$

$$U^T dW V = U^T dU\Sigma + d\Sigma + \Sigma dV^T V$$

## Распространение градиента через SVD



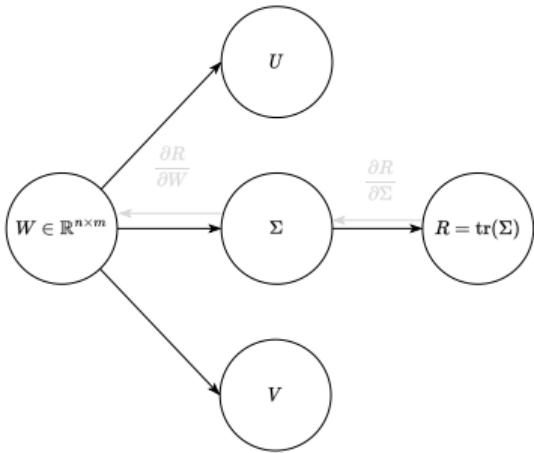
2. Обратите внимание, что  $U^T U = I \rightarrow dU^T U + U^T dU = 0$ . Но также  $dU^T U = (U^T dU)^T$ , что фактически означает, что матрица  $U^T dU$  является антисимметричной:

$$(U^T dU)^T + U^T dU = 0 \quad \rightarrow \quad \text{diag}(U^T dU) = (0, \dots, 0)$$

Та же логика может быть применена к матрице  $V$  и

$$\text{diag}(dV^T V) = (0, \dots, 0)$$

# Распространение градиента через SVD



2. Обратите внимание, что  $U^T U = I \rightarrow dU^T U + U^T dU = 0$ . Но также  $dU^T U = (U^T dU)^T$ , что фактически означает, что матрица  $U^T dU$  является антисимметричной:

$$(U^T dU)^T + U^T dU = 0 \quad \rightarrow \quad \text{diag}(U^T dU) = (0, \dots, 0)$$

Та же логика может быть применена к матрице  $V$  и

$$\text{diag}(dV^T V) = (0, \dots, 0)$$

3. В то же время, матрица  $d\Sigma$  является диагональной, что означает (смотрите 1.) что

$$\text{diag}(U^T dWV) = d\Sigma$$

Здесь на обеих сторонах мы имеем диагональные матрицы.

# Распространение градиента через SVD



4. Теперь мы можем разложить дифференциал функции потерь как функцию  $\Sigma$  - такие проблемы возникают в ML задачах, где мы должны ограничить ранг матрицы:

$$\begin{aligned} dL &= \left\langle \frac{\partial L}{\partial \Sigma}, d\Sigma \right\rangle \\ &= \left\langle \frac{\partial L}{\partial \Sigma}, \text{diag}(U^T dWV) \right\rangle \\ &= \text{tr} \left( \frac{\partial L}{\partial \Sigma}^T \text{diag}(U^T dWV) \right) \end{aligned}$$

# Распространение градиента через SVD



5. Как только мы имеем диагональные матрицы внутри произведения, след диагональной части матрицы будет равен следу всей матрицы:

$$\begin{aligned} dL &= \text{tr} \left( \frac{\partial L}{\partial \Sigma} \text{diag}(U^T dWV) \right) \\ &= \text{tr} \left( \frac{\partial L}{\partial \Sigma} U^T dWV \right) \\ &= \left\langle \frac{\partial L}{\partial \Sigma}, U^T dWV \right\rangle \\ &= \left\langle U \frac{\partial L}{\partial \Sigma} V^T, dW \right\rangle \end{aligned}$$

# Распространение градиента через SVD



6. Наконец, используя другую параметризацию дифференциала

$$\left\langle U \frac{\partial L}{\partial \Sigma} V^T, dW \right\rangle = \left\langle \frac{\partial L}{\partial W}, dW \right\rangle$$

$$\frac{\partial L}{\partial W} = U \frac{\partial L}{\partial \Sigma} V^T,$$

Этот результат позволяет нам связать градиенты  $\frac{\partial L}{\partial W}$  и  $\frac{\partial L}{\partial \Sigma}$ .

## Вычислительный эксперимент с JAX

Давайте убедимся численно, что мы правильно вычислили производные в задачах 2-3 🖨



## Архитектура прямого распространения

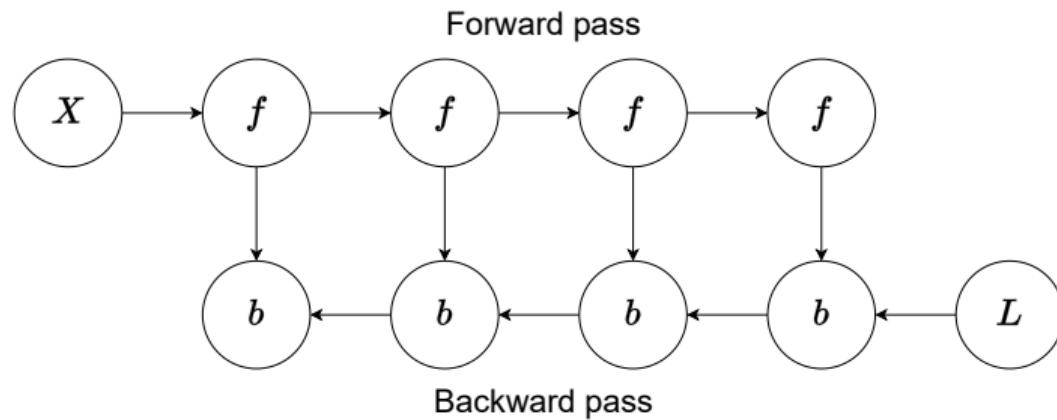


Рис. 9: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Активации отмечены  $f$ . Градиент функции потерь по отношению к активациям и параметрам отмечен  $b$ .

# Архитектура прямого распространения

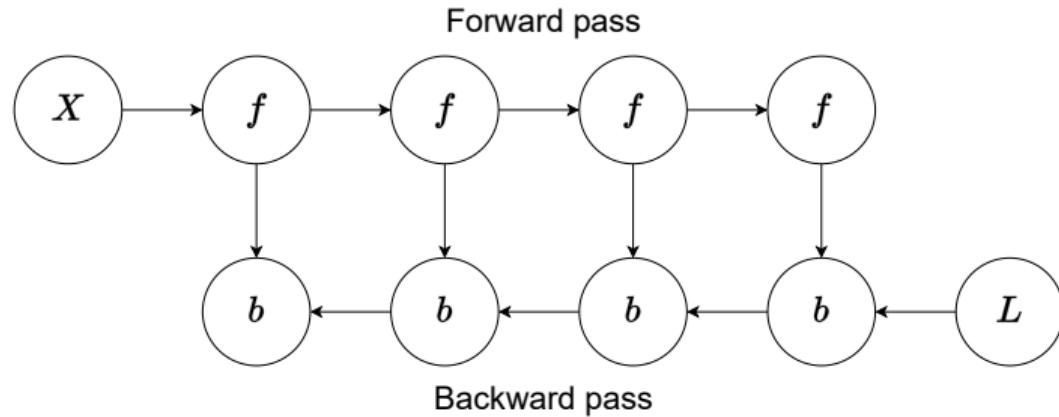


Рис. 9: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Активации отмечены  $f$ . Градиент функции потерь по отношению к активациям и параметрам отмечен  $b$ .

## ! Важное уведомление

Результаты, полученные для узлов  $f$ , необходимы для вычисления узлов  $b$ .

## Обычное обратное распространение

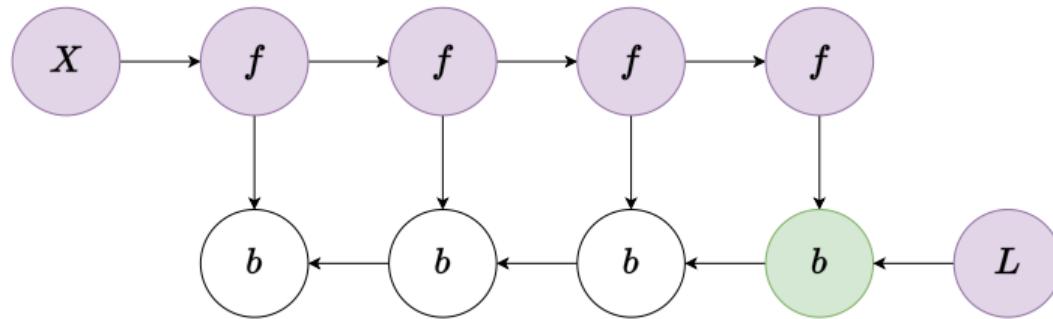


Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

## Обычное обратное распространение

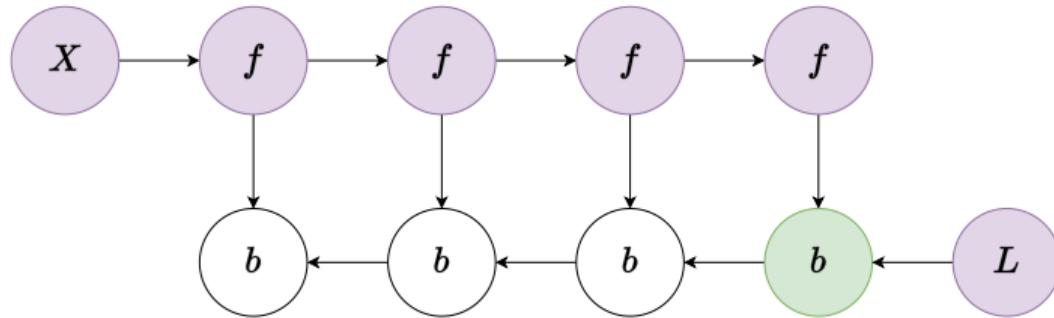


Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Все активации  $f$  хранятся в памяти после прямого прохода.

## Обычное обратное распространение



Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Все активации  $f$  хранятся в памяти после прямого прохода.

## Обычное обратное распространение

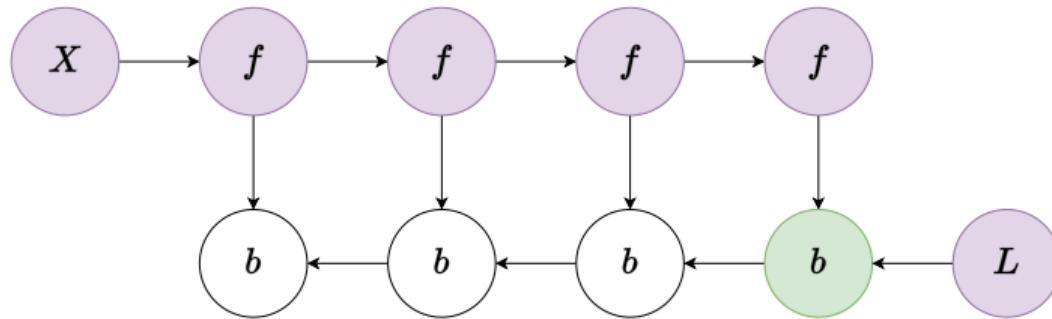


Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Все активации  $f$  хранятся в памяти после прямого прохода.
- Оптимально с точки зрения вычислений: оно вычисляет каждый узел только один раз.

## Обычное обратное распространение



Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Все активации  $f$  хранятся в памяти после прямого прохода.
- Оптимально с точки зрения вычислений: оно вычисляет каждый узел только один раз.

## Обычное обратное распространение

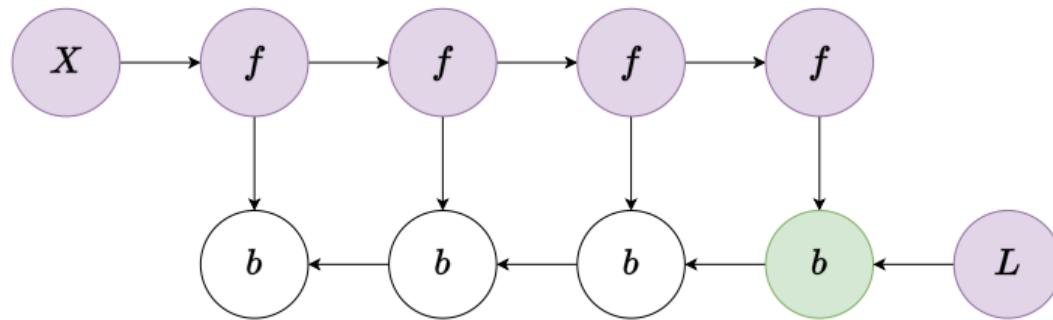


Рис. 10: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Все активации  $f$  хранятся в памяти после прямого прохода.
- Оптимально с точки зрения вычислений: оно вычисляет каждый узел только один раз.
- Высокое использование памяти. Использование памяти растет линейно с количеством слоев в нейронной сети.

## Ограниченнное по памяти обратное распространение



Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

## Ограниченнное по памяти обратное распространение



Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Каждая активация  $f$  пересчитывается при необходимости.

## Ограниченнное по памяти обратное распространение



Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Каждая активация  $f$  пересчитывается при необходимости.

## Ограниченнное по памяти обратное распространение



Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $L$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Каждая активация  $f$  пересчитывается при необходимости.
- Оптимально с точки зрения памяти: нет необходимости хранить все активации в памяти.

## Ограниченнное по памяти обратное распространение



Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $L$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Каждая активация  $f$  пересчитывается при необходимости.
- Оптимально с точки зрения памяти: нет необходимости хранить все активации в памяти.

## Ограниченнное по памяти обратное распространение

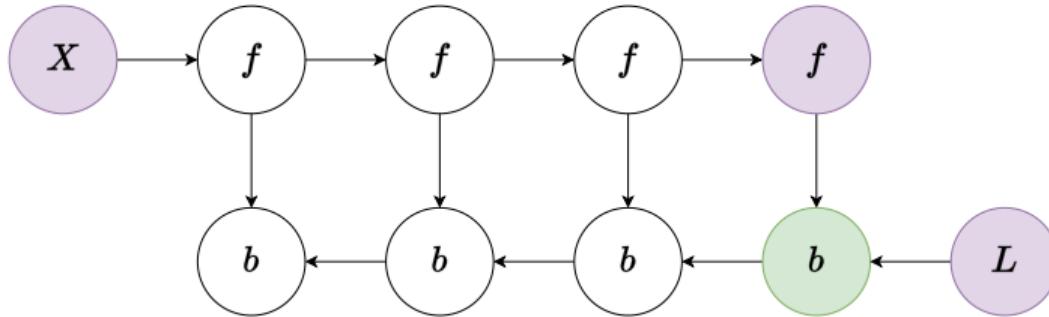


Рис. 11: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Каждая активация  $f$  пересчитывается при необходимости.
- Оптимально с точки зрения памяти: нет необходимости хранить все активации в памяти.
- Вычислительно неэффективно. Количество оценок узлов масштабируется как  $n^2$ , в то время как в обычном обратном распространении оно масштабируется как  $n$ : каждый из  $n$  узлов пересчитывается порядка  $n$  раз.

## Контрольные точки обратного распространения

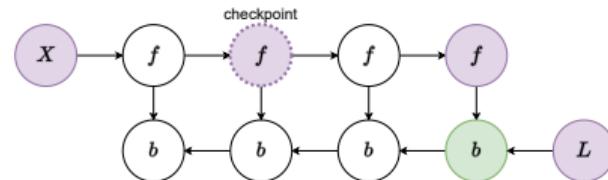


Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

## Контрольные точки обратного распространения

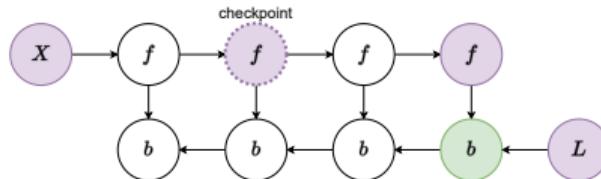


Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Компромисс между **обычным** и **ограниченным по памяти** подходами. Стратегия состоит в том, чтобы отметить подмножество активаций нейронной сети как контрольные точки, которые будут храниться в памяти.

## Контрольные точки обратного распространения

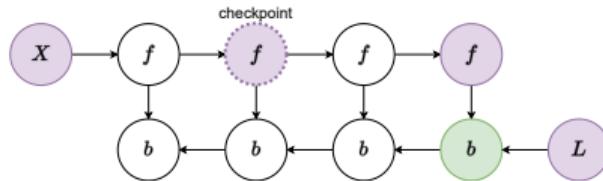


Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Компромисс между **обычным** и **ограниченным по памяти** подходами. Стратегия состоит в том, чтобы отметить подмножество активаций нейронной сети как контрольные точки, которые будут храниться в памяти.

## Контрольные точки обратного распространения

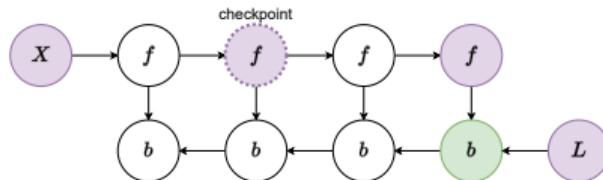


Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Компромисс между **обычным** и **ограниченным по памяти** подходами. Стратегия состоит в том, чтобы отметить подмножество активаций нейронной сети как контрольные точки, которые будут храниться в памяти.
- Быстрее пересчитывание активаций  $f$ . Мы только пересчитываем узлы между узлом  $b$  и последней контрольной точкой, предшествующей ему, при вычислении этого узла  $b$  во время обратного распространения.

## Контрольные точки обратного распространения

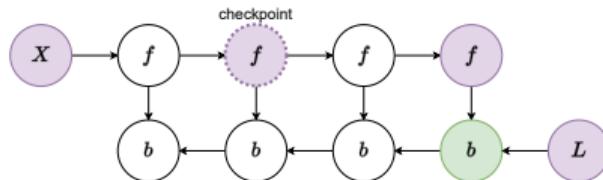


Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Компромисс между **обычным** и **ограниченным по памяти** подходами. Стратегия состоит в том, чтобы отметить подмножество активаций нейронной сети как контрольные точки, которые будут храниться в памяти.
- Быстрее пересчитывание активаций  $f$ . Мы только пересчитываем узлы между узлом  $b$  и последней контрольной точкой, предшествующей ему, при вычислении этого узла  $b$  во время обратного распространения.

## Контрольные точки обратного распространения



Рис. 12: Вычислительный граф для получения градиентов для простого прямого распространения нейронной сети с  $n$  слоями. Фиолетовый цвет указывает узлы, которые хранятся в памяти.

- Компромисс между **обычным** и **ограниченным по памяти** подходами. Стратегия состоит в том, чтобы отметить подмножество активаций нейронной сети как контрольные точки, которые будут храниться в памяти.
  - Быстрее пересчитывание активаций  $f$ . Мы только пересчитываем узлы между узлом  $b$  и последней контрольной точкой, предшествующей ему, при вычислении этого узла  $b$  во время обратного распространения.
  - Использование памяти зависит от количества контрольных точек. Более эффективно, чем **обычный** подход.

# Визуализация контрольных точек обратного распространения

Анимация вышеуказанных подходов 

Пример использования контрольных точек градиентов 

## Оценка следа Гессиана<sup>1</sup>

Этот пример иллюстрирует оценку следа Гессиана нейронной сети с использованием метода Hutchinson, который является алгоритмом для получения такой оценки из матрично-векторных произведений:

Пусть  $X \in \mathbb{R}^{d \times d}$  и  $v \in \mathbb{R}^d$  случайный вектор такой, что  $\mathbb{E}[vv^T] = I$ . Тогда,

$$\text{Tr}(X) = \mathbb{E}[v^T X v] \approx \frac{1}{V} \sum_{i=1}^V v_i^T X v_i.$$

Пример использования оценки следа  
Гессиана Hutchinson 



Рис. 13: Несколько запусков оценки следа Гессиана Hutchinson, инициализированных при разных случайных начальных значениях.

<sup>1</sup>A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines - M.F. Hutchinson, 1990