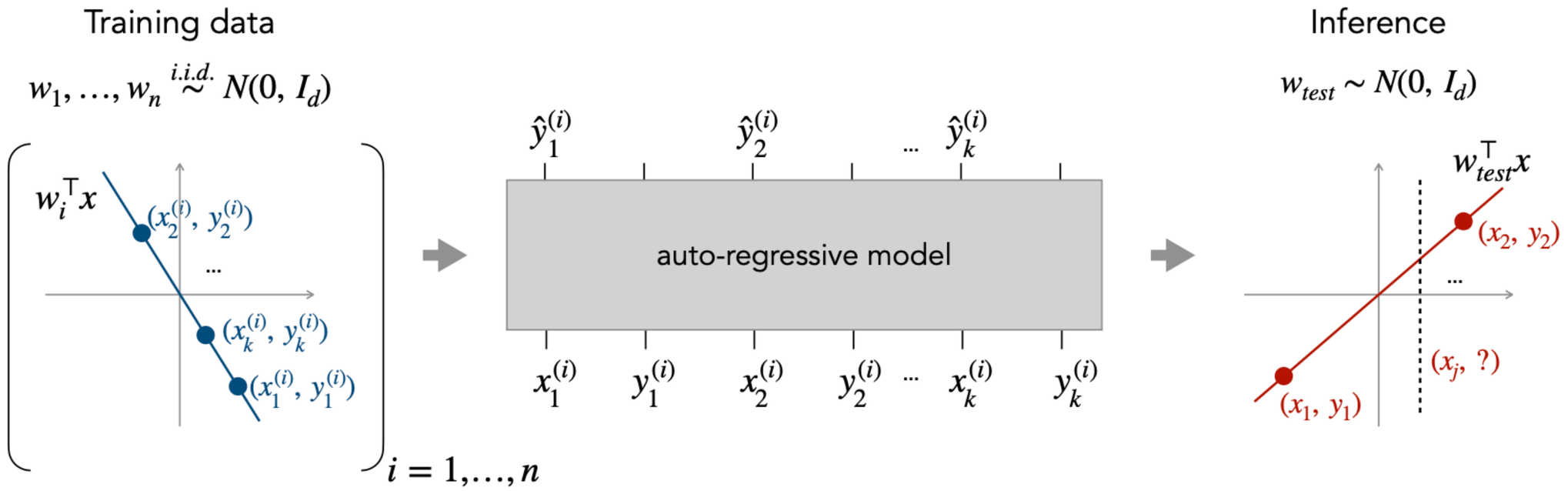
# Что трансформеры могут выучить из контекста?

# Основная идея статьи

- После выхода GPT-3 стало понятно, что модель умеет в in-context learning
- Хочется понять, “скормили” ли модели много данных и она видела ответы на вопросы из промптов или действительно “понимает” структуру данных
- В результате авторы пришли к выводу, что “понимает”

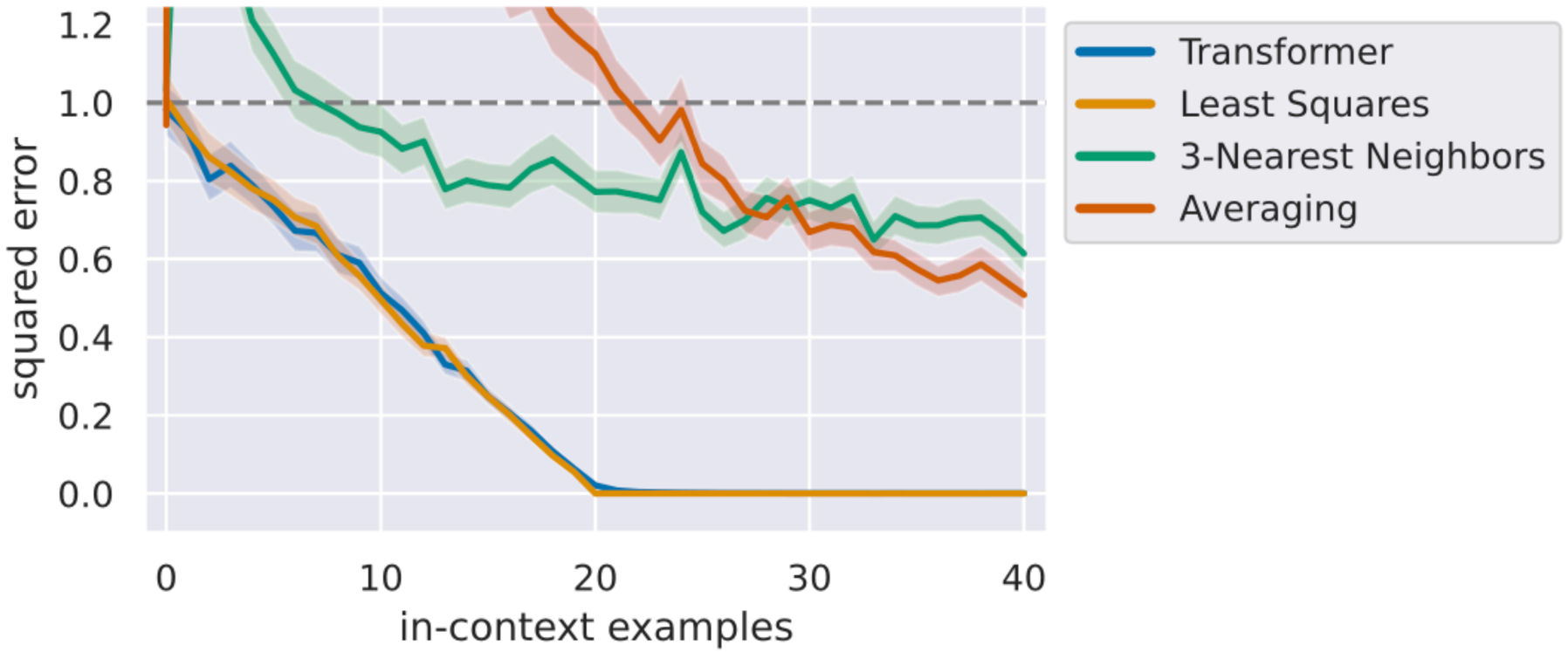
# Первый эксперимент



# Первый эксперимент

- Обучали GPT-2 на 9.5M параметров
- Получили качество примерно равное оценке наименьших квадратов (оптимальной для данной задачи)
- Во время обучения модель видела 32M различных векторов весов, если для каждого тестового примера использовать оптимальный вектор из этих, то ошибка составила бы 0.2. Трансформер же показывает ошибку 0.0006. Что говорит о понимании структуры данных, а не о запоминании увиденных примеров

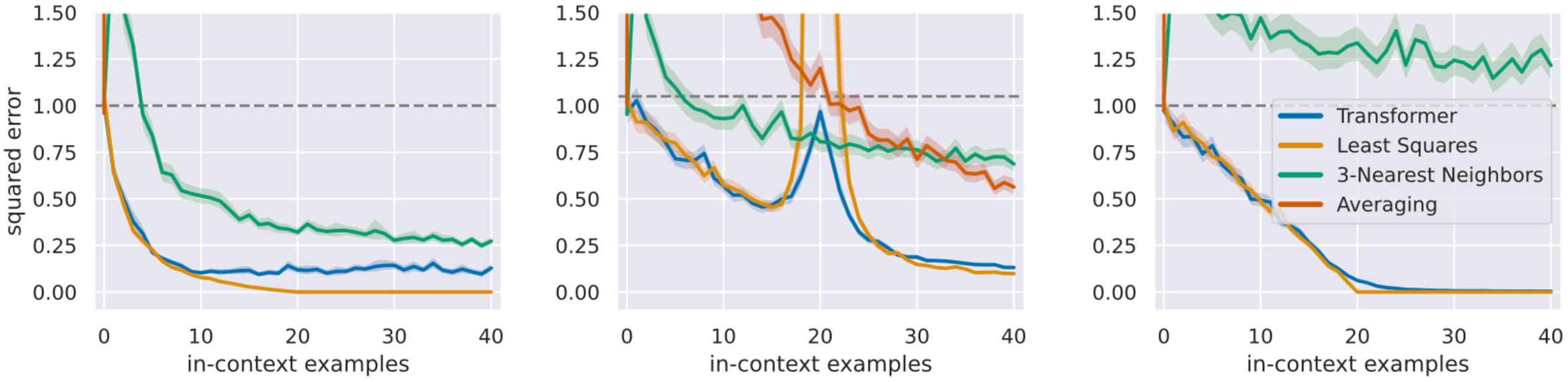
# Первый эксперимент



# Первый эксперимент

- До этого обучались и тестировались на одном и том же распределении
- Посмотрим сильно ли упадут результаты если тестироваться на другом
- Оказывается, что всё ещё получаются хорошие результаты

# Первый эксперимент

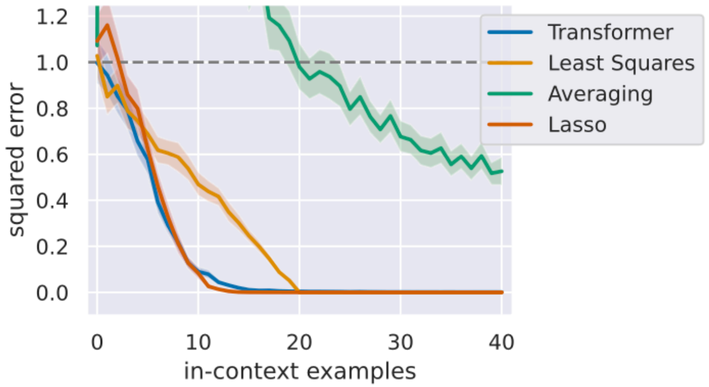



(a) Skewed covariance

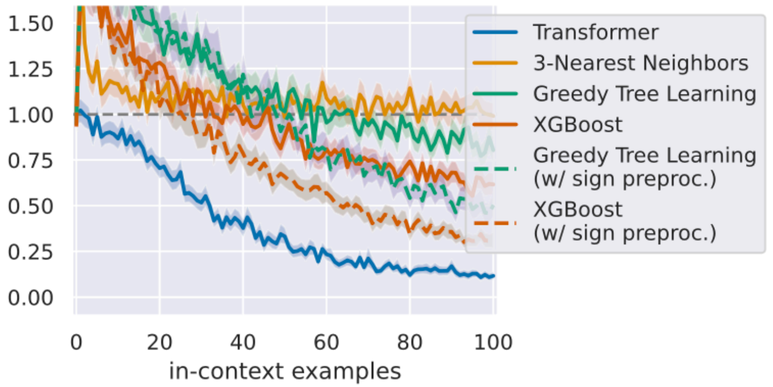
(b) Noisy linear regression

(c) Different orthants

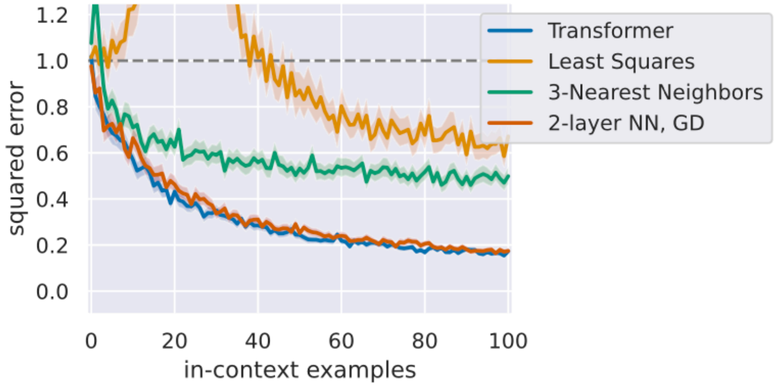
# Много других функций



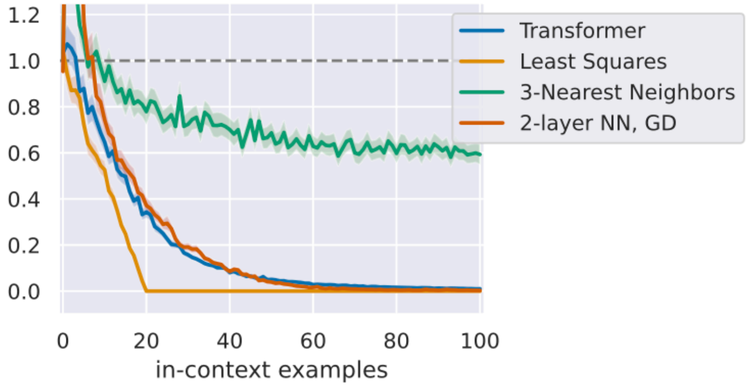

(a) Sparse linear functions




(b) Decision trees




(c) 2-layer NN

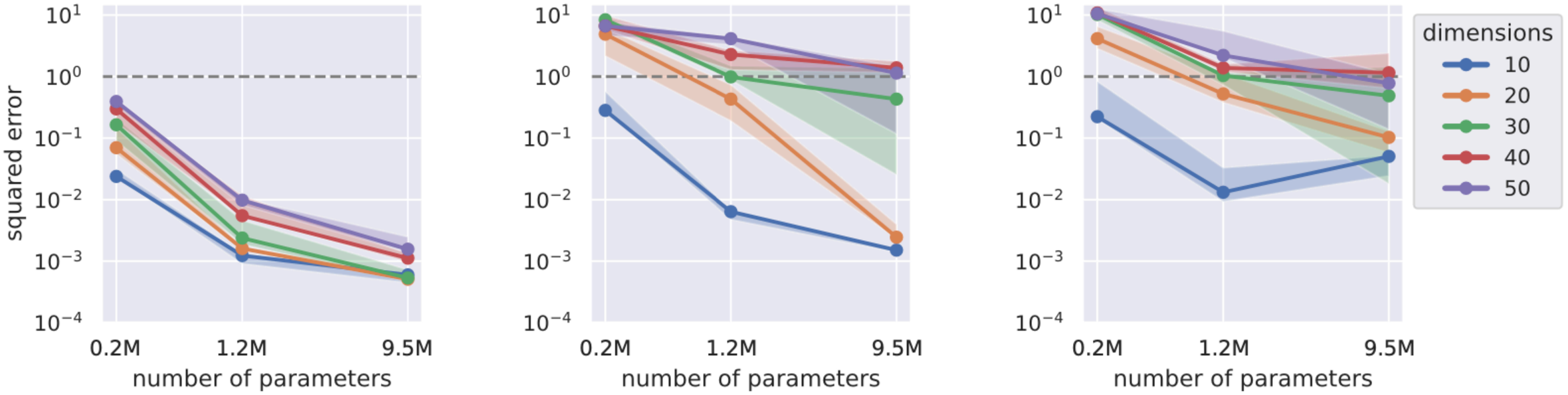



(d) 2-layer NN, eval on linear functions

# Много других функций

- С нейронками всё понятно
- Деревья используются простые: полное бинарное глубины 4. В качестве условия для не листовых вершин мы смотрим больше или меньше соответствующая координата нуля

# Зависимость от числа параметров



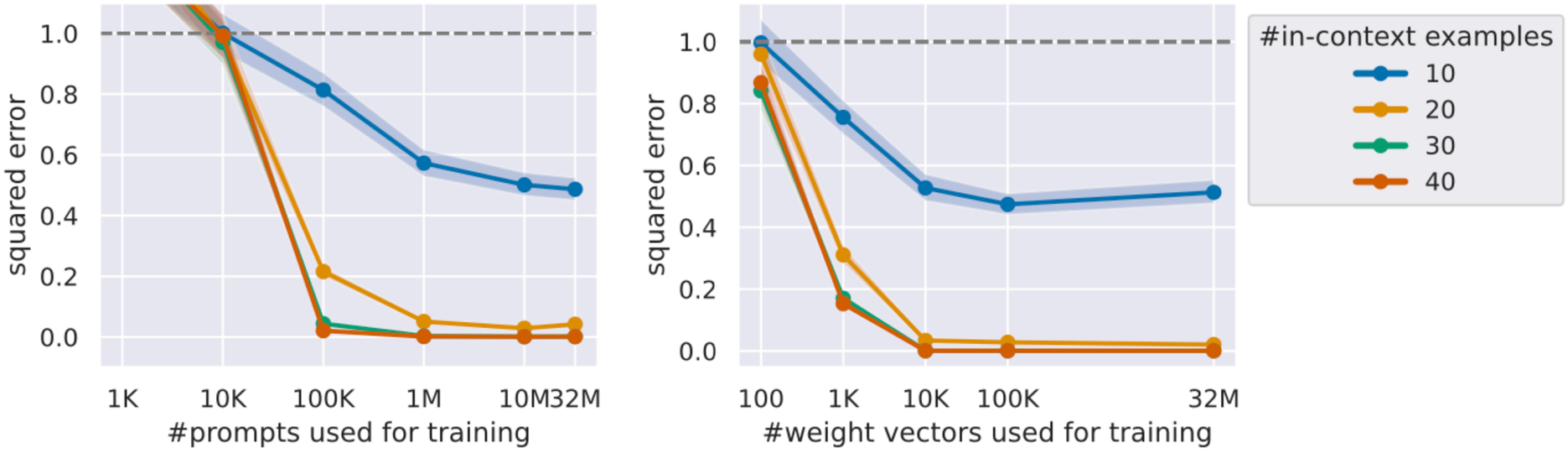

(a) Standard

(b) Different orthants

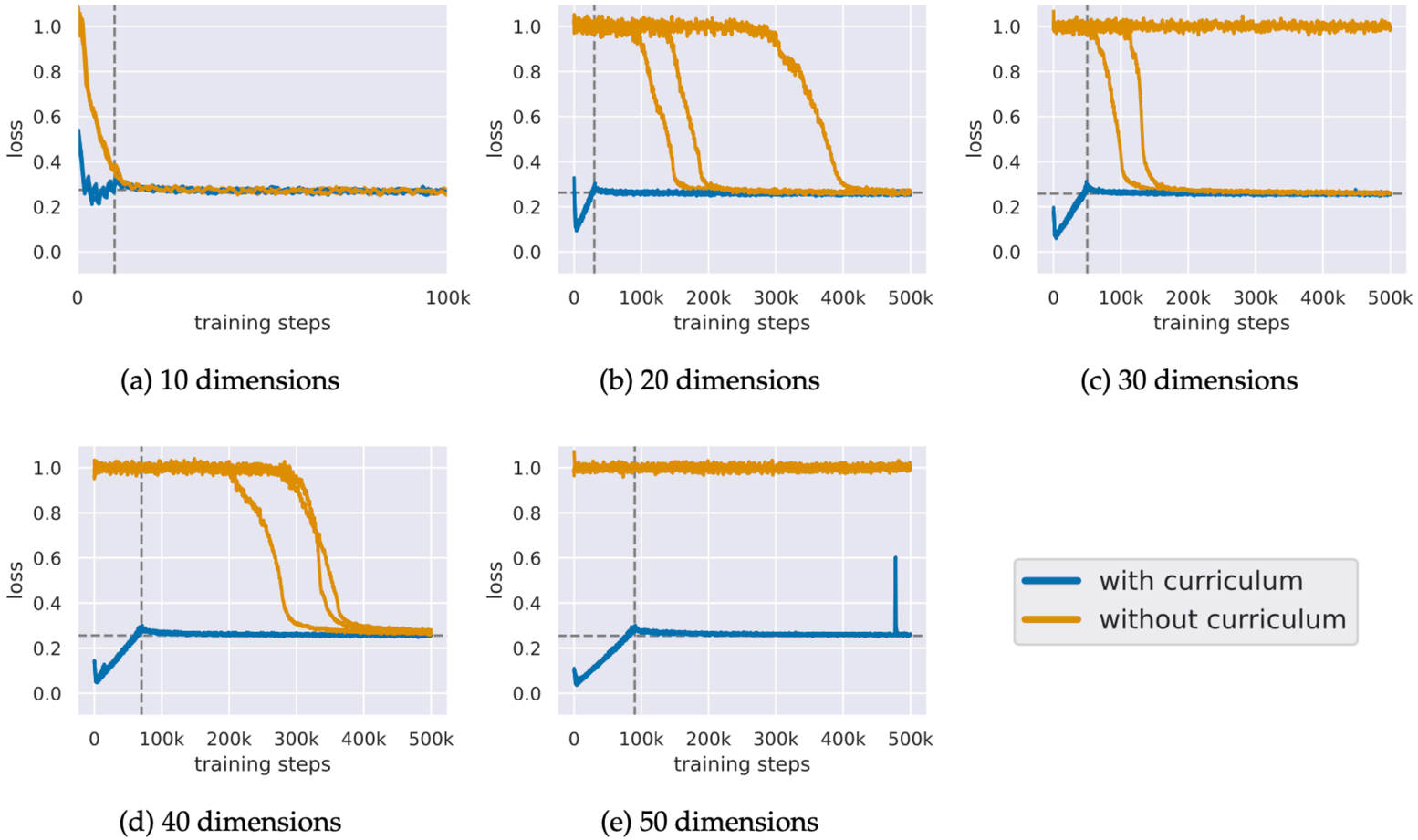
(c) Skewed covariance

# А сколько же обучать?

- Оказывается, что относительно недолго. Для хороших результатов достаточно 100k промптов или 1к векторов весов.



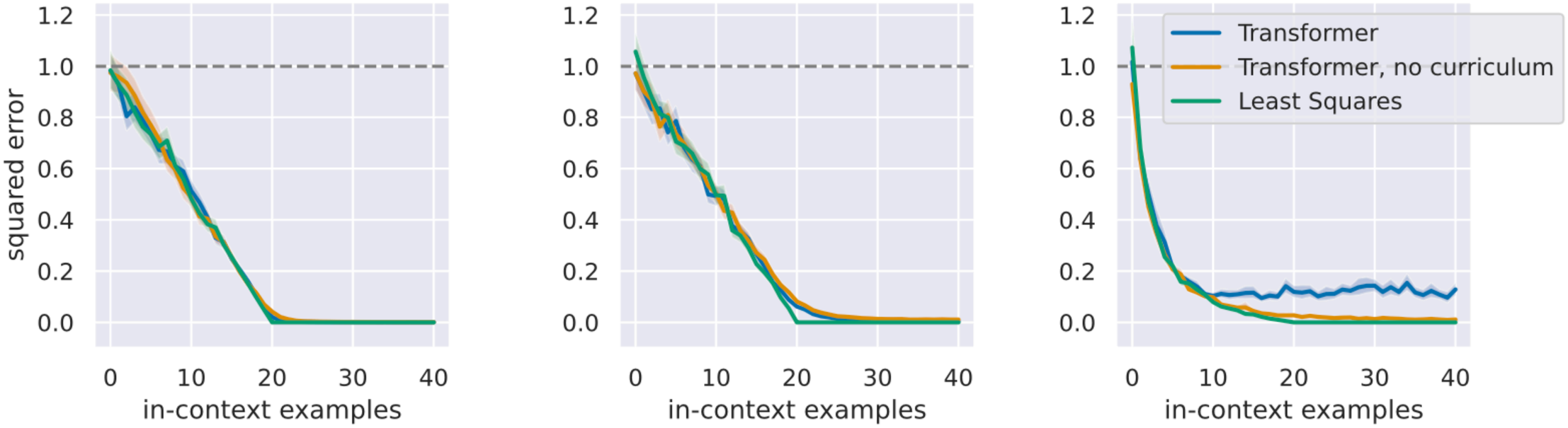
# Важность расписания




(a) 10 dimensions (b) 20 dimensions (c) 30 dimensions

(d) 40 dimensions (e) 50 dimensions

# Важность расписания

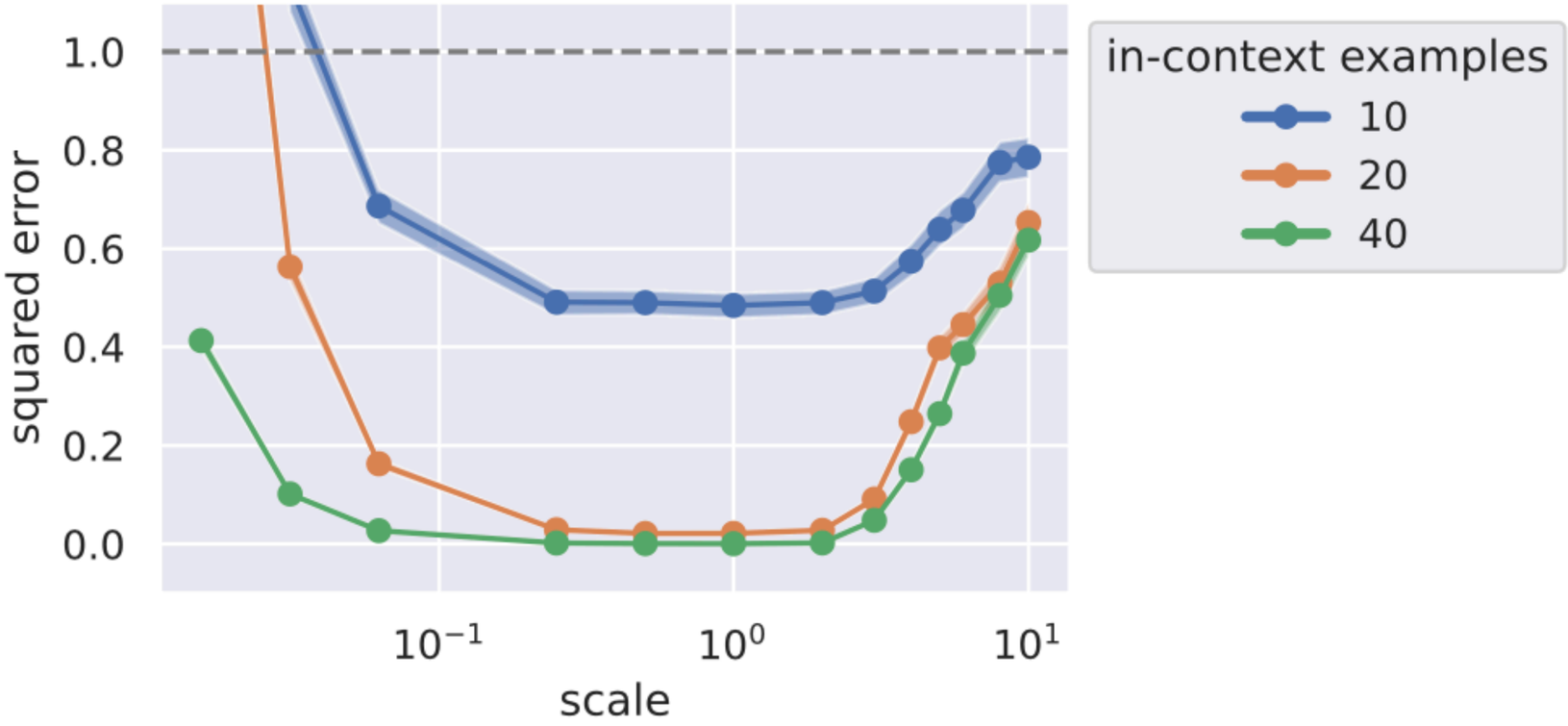



(a) standard

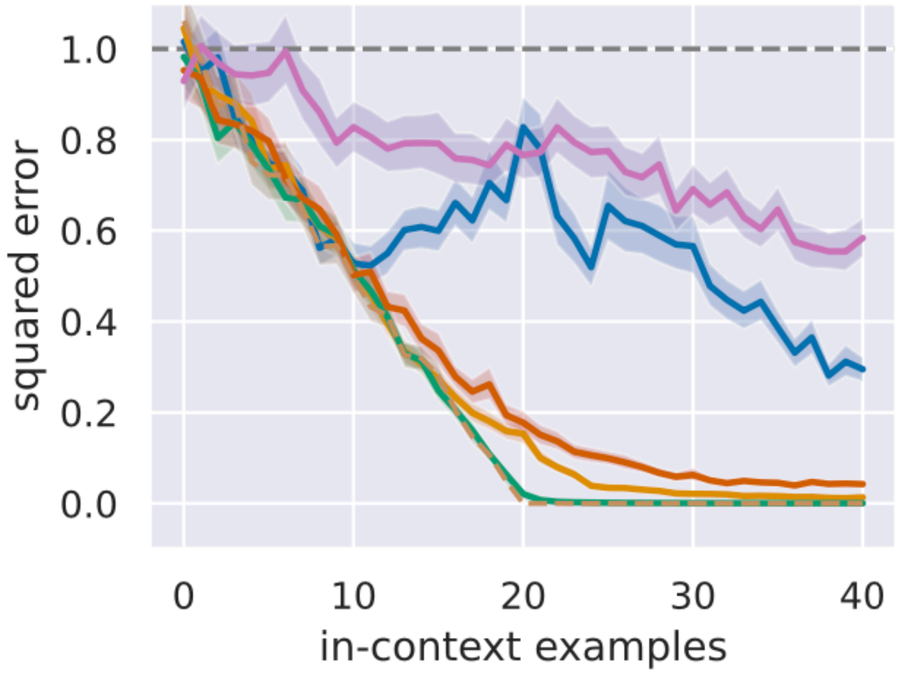
(b) different orthants

(c) skewed

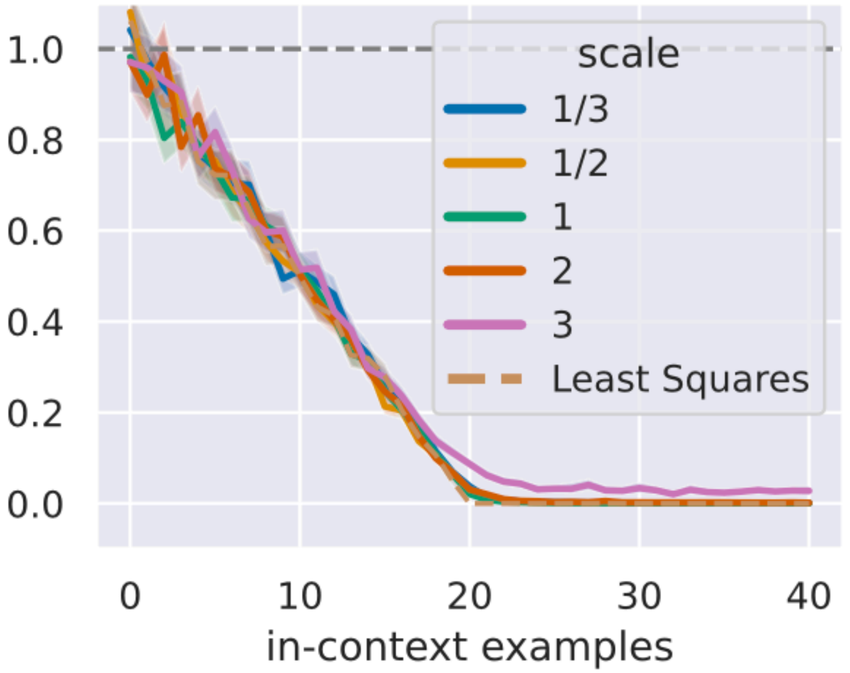
# Скейлинг


squared error
1.0
0.8
0.6
0.4
0.2
0.0
$10^{-1}$
$10^{0}$
$10^{1}$
scale
in-context examples
10
20
40

# Скейлинг




(a) scaled $x$, Transformer




(b) scaled $w$, Transfomer

# Итог

- Трансформер действительно может выучить функцию из контекста