

# ЧТО ИСПОЛЬЗУЕМ

- Шрифт для заголовков — [скачать](#)
- Не в заголовках используем — Lab Grotesque K Regular (Основной текст)
- Основные цвета — ff007f 9700ff
- Дополнительные цвета темы можем использовать для графиков и сценариев, где нужно показать сравнение с 2-мя и более цветами



# ЗАЧЕМ В ML НУЖНЫ ПРОИЗВОДНАЯ, ГРАДИЕНТ И АЛГОРИТМЫ

Александр Панкратов

Разработчик машинного обучения





# ЧТО ОБСУДИМ

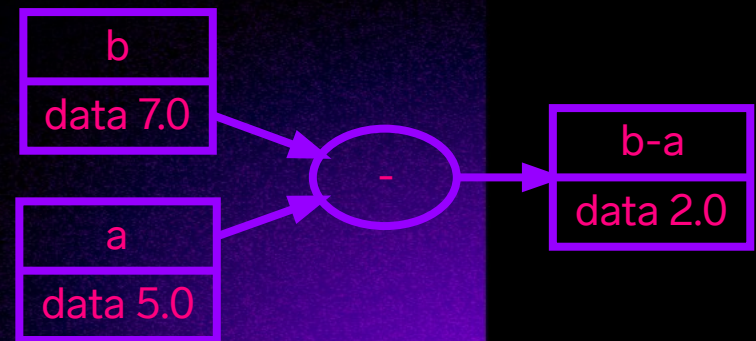
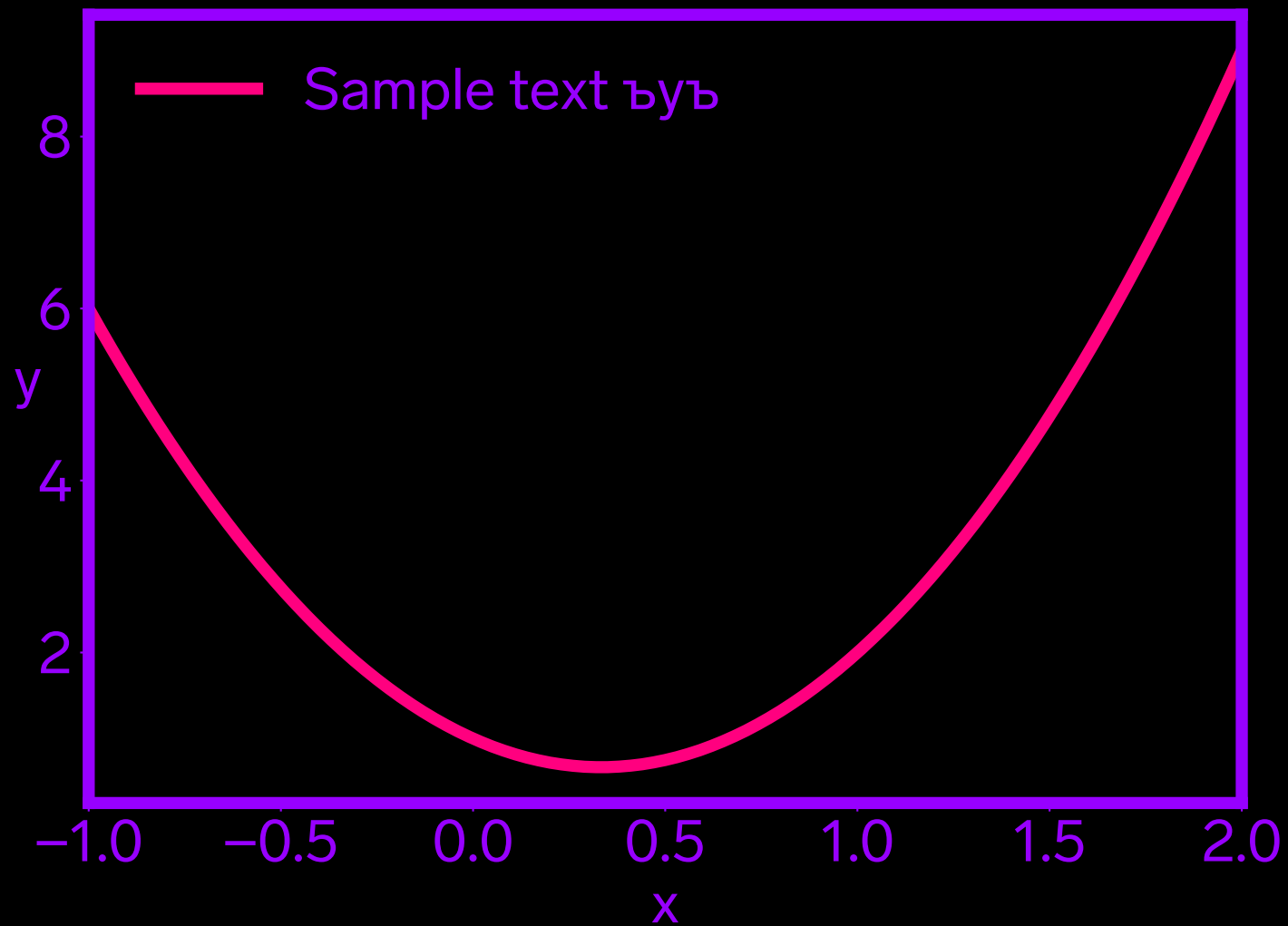
- Чуть подробнее, чем это обычно принято, посмотрим как обучается нейросеть.<sup>23</sup>
- Для чего в точности дата-саентистам GPU, и не лучше ли Контуру купить чугунный мост вместо GPU-серверов?
- Где и зачем в МЛ встречается динамическое программирование.

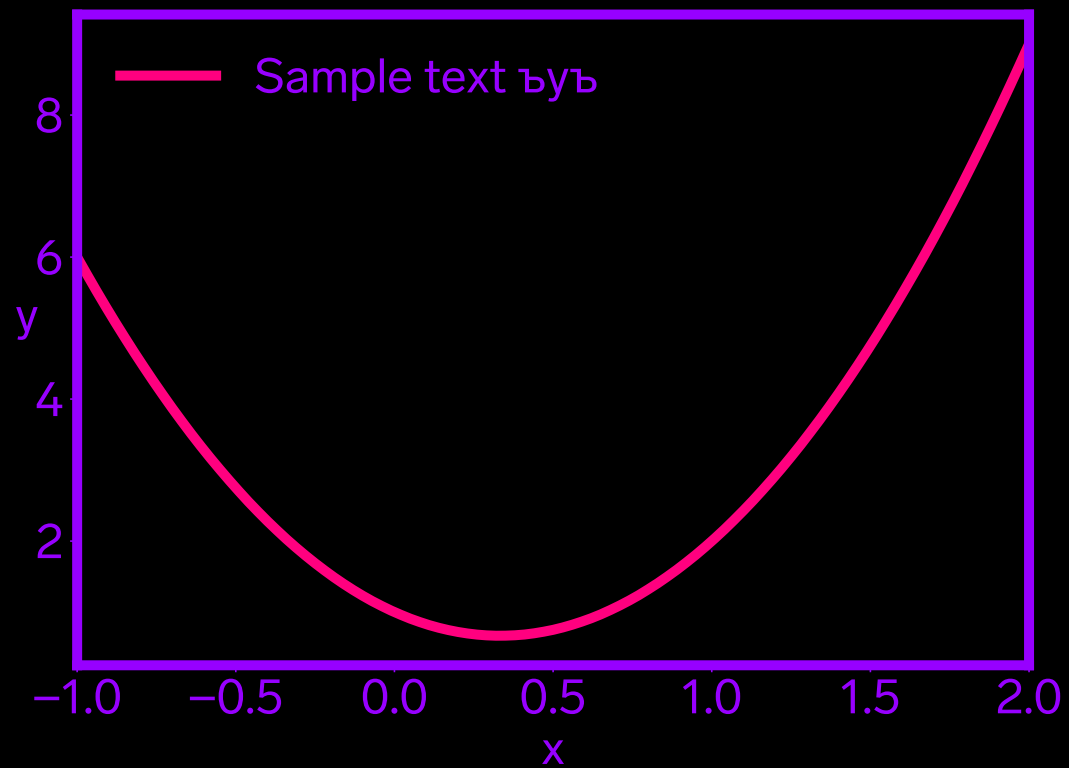
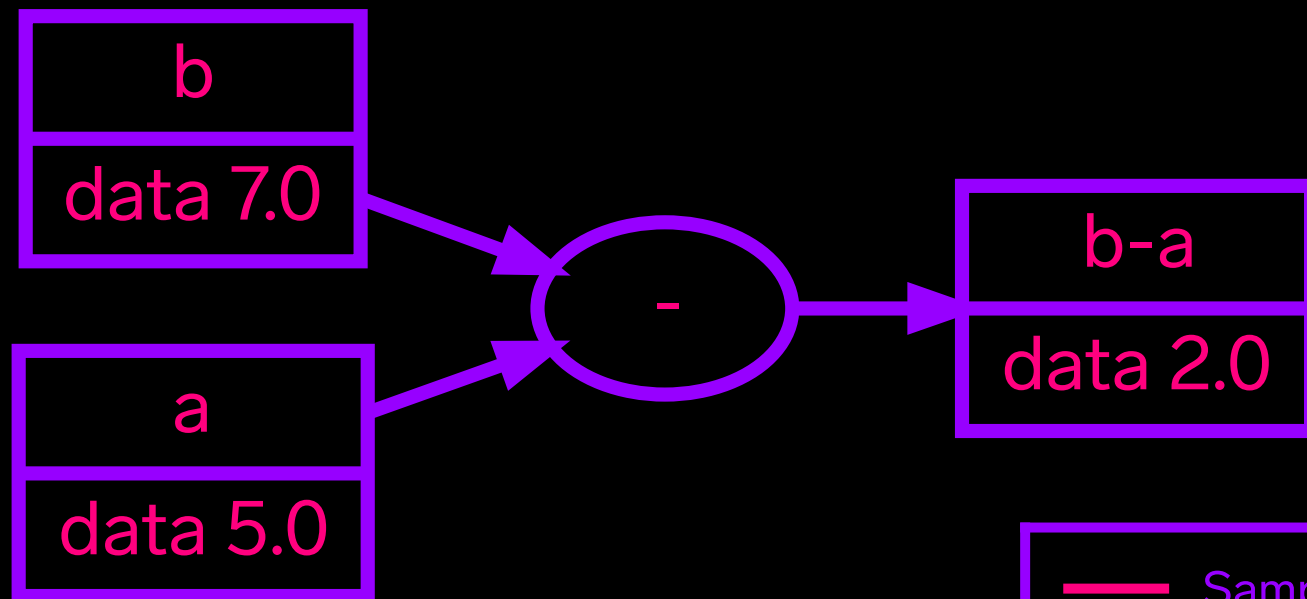


ПРОИЗВОДНАЯ И ЕЁ ДРУЗЬЯ

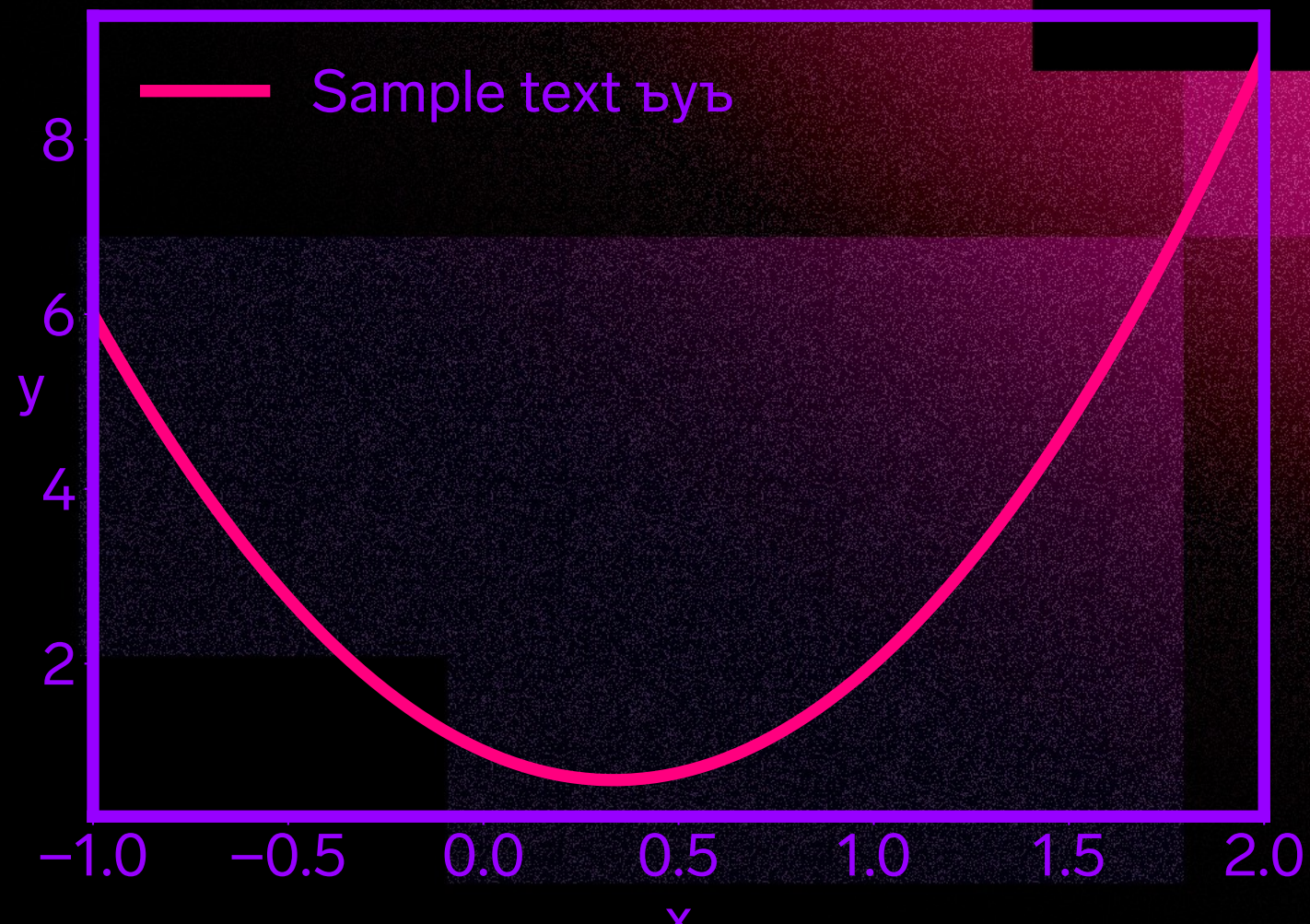
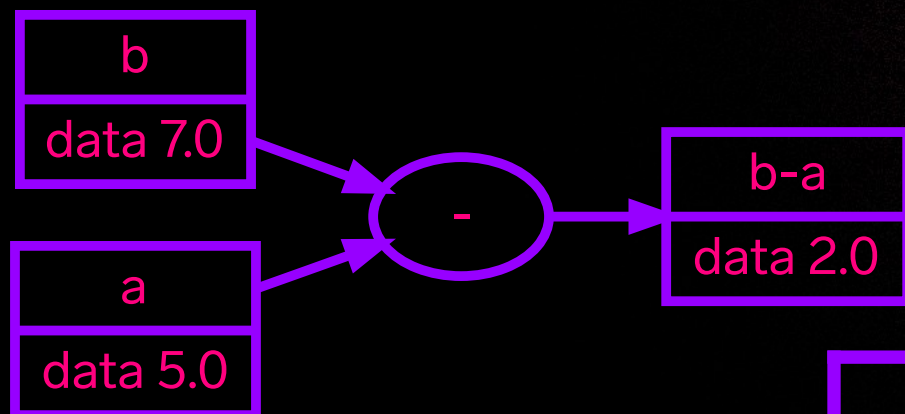


# ПРОИЗВОДНАЯ: ВСПОМНИТЬ ВСЁ











GPU, CUDA И БОЛЬШИЕ МАТРИЦЫ



# The nine circles of matmul

for row in A      triply nested for loop      "lol how is  
for col in B      in memory managed language      this hard?"  
for pair in (row,col)

realizing it's  $O(N^3)$       triply nested for loop      "wow it's 10,000x faster  
in non memory managed language      this is the end right?"

lowered time complexity matmuls      "wow it's  $O(N^{2.something})$   
like strassen's algorithm      now, this is great!"  
"wow multiplications are expensive"      multiplication factoring trick

block partitioning      optimized triply nested for loop      loop strength  
in non memory managed language  
realizing that lowered time complexity is garbage      SIMD      cache hierarchies      cache locality

single threaded happy times stop here. now everything is multiprocessed

autotuning      multithreaded matmuls      cache coherence  
deadlocks      race conditions  
killing children      load balancing      mutexes and semaphores

massively parallel matmuls (GPUs)  
hyperdimensional cache/thread      warp scheduling  
hierarchies      CUDA      lol you're learning a new  
batched matmuls      computing architecture now, sorry bucko

MASSIVELY PARALLEL MATMULS (distributed)  
heterogeneous computing      MapReduce      sharding matrices  
benchmarks cost \$\$\$      fault tolerance  
hardware starts failing      network topology

memory alignment      Arcane optimizations  
kernel search      hardware specific hand tuning  
mixed precision arithmetic      leveraging algebras  
sparse matmuls  
square vs rectangular vs triangular optimizations      fused kernels



А ЧТО ЕСЛИ ДАЖЕ GRC НЕ ХВАТАЕТ?











