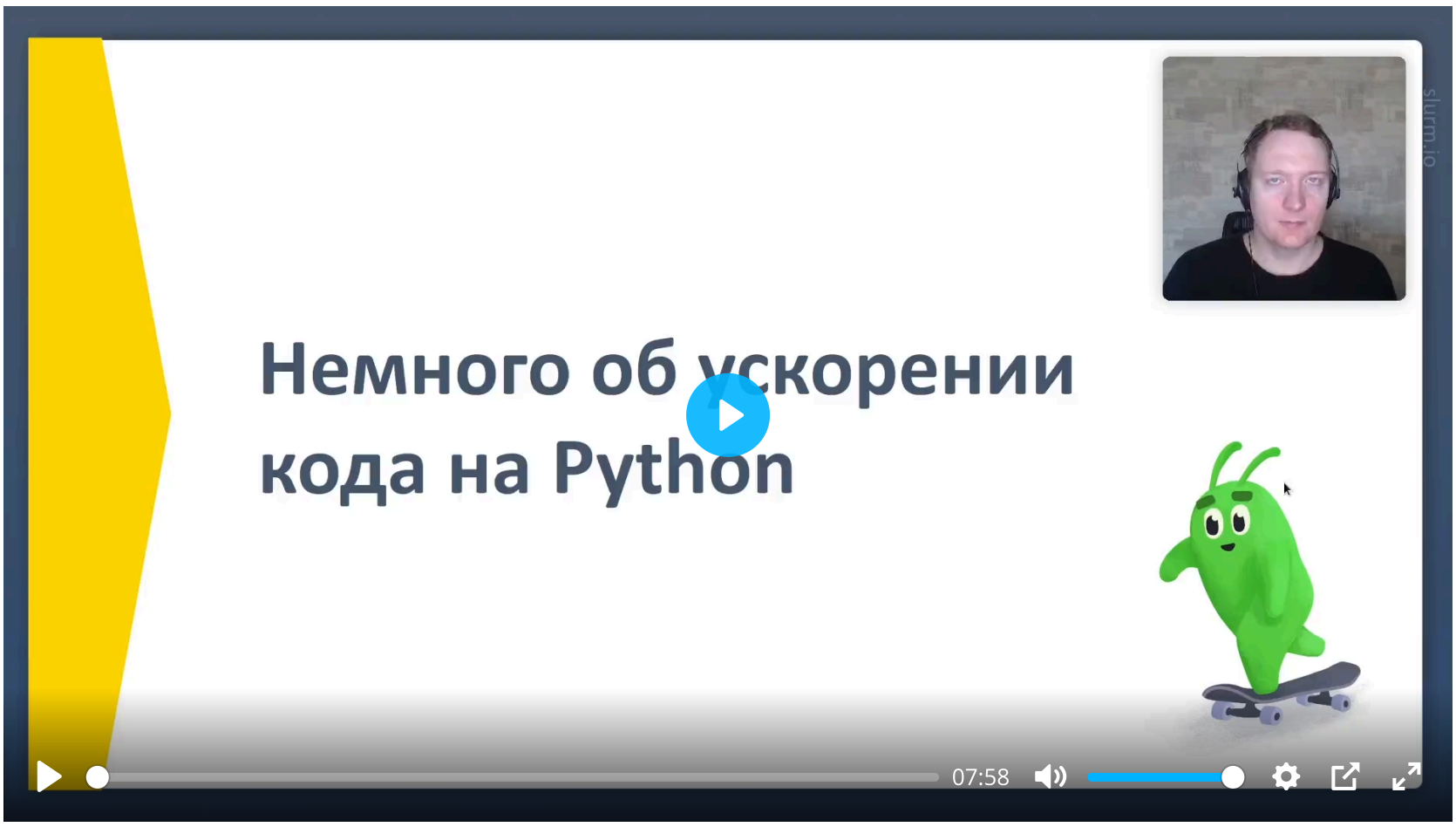
2.2 Инструменты экосистемы Python для работы с данными



Текстовая расшифровка видео:

НЕМНОГО ОБ УСКОРЕНИИ КОДА

План:

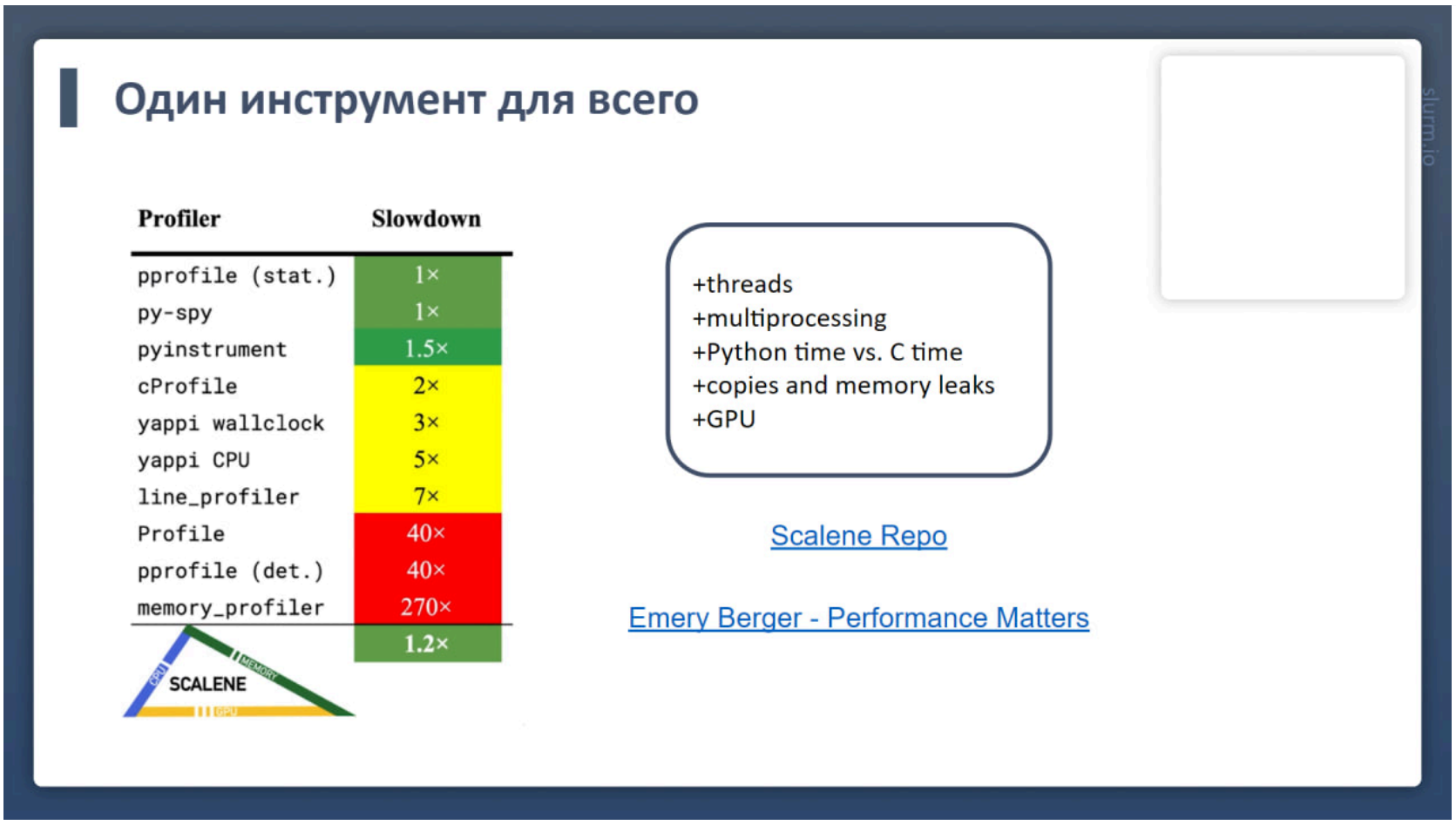
- Один инструмент для всего;
- Just In Time – компиляция;
- Numba

Один инструмент для всего

Один из способов ускорения на Python – профилирование.

Инструмент для ускорения:

- Scalene Repo



Плюсы:

- Быстрый;
- Профилирует время выполнения;
- Профилирует память;
- Поддерживает время сравнения кода;
- Имеет базовую поддержку профилирования на JPU.

Just In Time – компиляция

У нас есть ряд языков программирования, которые запускаются через дополнительную виртуальную машину (например, Java, Python и т.д.).

Запуск таких приложений происходит следующим образом:

- Приложение компилируется в Bytecode виртуальной машины;
- Виртуальная машина исполняет Bytecod и уже знает, как его разложить на архитектуру.

В процессе выполнения Bytecod виртуальная машина может собирать статистическую информацию о деятельности этого кода и пытаться оптимизировать в процессе некоторые его аспекты (например, кэшировать определенные значения, хранить в регистровых процессорах хранить и т.д.).

Многие современные инструменты используют реализации JIT на основе LLVM. Это полноценная виртуальная машина, позволяющая компиляторам и рантаймам применять различные оптимизации при выполнении кода.

Нюанс: стандартный дистрибутив Python (sipayton) подобного не умеет. В стандартной поставке отсутствует JIT. Существует отдельный for, который называется «PyPy», использующий LLVM для этого. Это дополнительный, отдельный мир, в котором есть свои недостатки.

Numba

Numba – проект, позволяющий ускорить код.

```
@njit(parallel=True)
def logistic_regression(Y, X, w, iterations):
  for i in range(iterations):
      w -= np.dot(((1.0 /
           (1.0 + np.exp(-Y * np.dot(X, w)))
           - 1.0) * Y), X)
  return w
```

Суть применения: мы описываем функцию, которая делает расчет; функцию оборачиваем в декоратор (он позволяет Numb'e в процессе выполнения смотреть на Bytecod и принимать решения о том, что к нему можно подключить).

На сложных кусках кода Numba умеет неплохо оптимизировать производительность.

Есть разработчик Марк Шеннон, который несколько лет назад представил план по тому, как ускорить Sipayton.

Можно заметить, что для новых версий Python уже есть поддержка в виде JIT.