

# Нейросетевой синтез текстур с трендами

Будамян Я. С.

Научный руководитель: к.т.н., доцент Грачев Е. А.

Москва, 2017 г.

# Введение

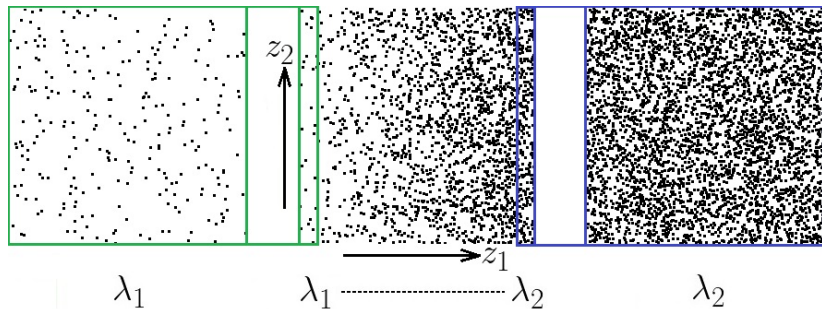
- ▶ Геологические среды часто имеют пространственно скоррелированные неоднородности, поэтому при решении проблемы генерирования геологоподобных сред возникает задача синтеза текстур с устойчивыми протяженными корреляциями.
- ▶ Цель работы: применение нейросетевых подходов для синтеза текстур с трендами, т.е. с устойчивым изменением некоторой статистической характеристики вдоль одного из направлений.

# Задача и модельные ограничения

Задача: рассмотреть синтез текстур из множества изображений с трендами, являющихся моделью среды, состоящей из отдельных частиц и удовлетворяющих ограничениям:

- ▶ Монохромные изображения 256 x 256 пикселей
- ▶ Изменяющимся свойством является интенсивность появления частиц  $\lambda$
- ▶ Тренд является линейным и направлен вдоль оси изображения  $z_1$ :  $\lambda = \lambda_{init} + kz_1$
- ▶ По оси  $z_2$  остается равномерное распределение частиц

# Пример входных данных



Пример изображения с трендом, фиксируемого двумя изображениями

# Математическая формализация

Математически задача синтеза текстур описывается с помощью вероятностной постановки задачи обучения:

- ▶ Рассматривается многомерное пространство  $X$ , содержащее множество всех изображений  $x$ :  $X = \{x\}$
- ▶ Есть обучающая выборка, состоящая из текстур с трендами  $D = \{x_i\}$ ,  $D \subset X$
- ▶ Считается, что  $D$  задает в  $X$  вероятностное распределение  $P_X : X \rightarrow [0, 1]$

# Математическая формализация

Таким образом задача синтеза текстуры из нужного множества сводится к синтезу случайного изображения  $x'$  из распределения, близкого к задаваемому обучающей выборкой:

$$P_{X'} \approx P_X, \quad x' \sim X'$$

# GAN

Генеративные состязательные сети (GAN - Generative Adversarial Networks) были придуманы в 2014 году и достигли больших успехов в задачах синтеза объектов из сложных распределений.

- ▶ Переформулируем:  $P_{X'} \approx P_X \Leftrightarrow \rho(P_{X'}, P_X) \longrightarrow \min_{P_{X'}}$
- ▶  $X' = g_\theta(\cdot) \Rightarrow \rho(g_\theta(\cdot), P_X) \longrightarrow \min_\theta$
- ▶ В качестве  $\rho$  можно использовать функцию потерь обученного классификатора

# GAN

Вводятся две нейросети:

- ▶  $d_{\zeta}(x)$  - классификатор для измерения расстояния, **дискриминатор**
- ▶  $g_{\theta}(x)$  - сеть, трансформирующая шум в элементы множества  $X'$ , **генератор**

Суть использования двух сетей состоит в том, что они обучаются совместно, конкурируя друг с другом.

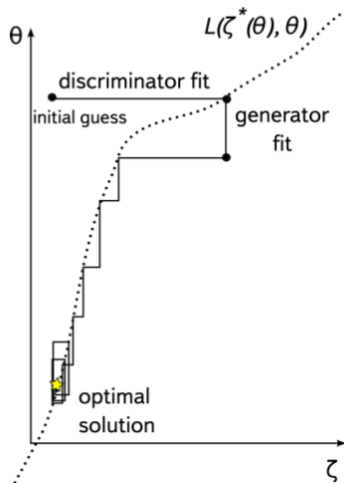
$$\theta^* = \arg \max_{\theta} \left[ \min_{\zeta} L(\zeta, \theta) \right]$$



# GAN

Процесс обучения сети GAN принимает следующий вид:

- ▶ Обучаем дискриминатор при фиксированном генераторе
- ▶ Обучаем генератор при фиксированном дискриминаторе
- ▶ Повторяем до сходимости параметров обеих моделей



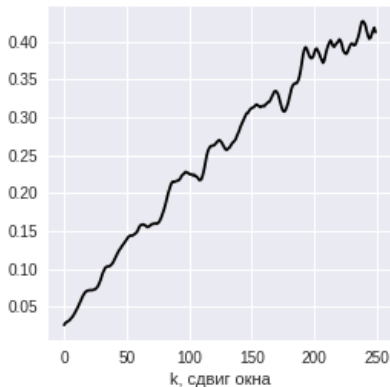
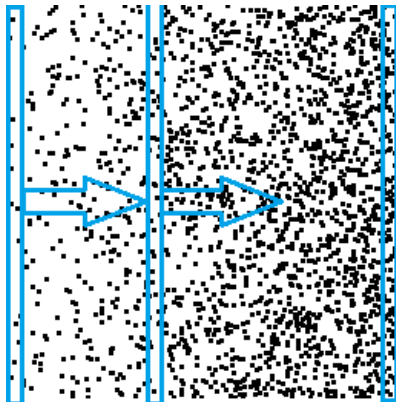
# Оценка качества синтеза текстур

Вводится специальная метрика, которая будет учитывать наличие в изображении тренда интенсивности частиц. Рассмотрим среднюю плотность черных пикселей в некотором окне  $\xi_k$ , и пройдем этим окном по изображению.

$$\xi_k = \frac{1}{Hw} \sum_{i=k}^{k+w} \sum_{j=0}^H \left| \frac{x(i,j) - 255}{255} \right|,$$
$$k = \overline{1, W - w}$$

# Оценка качества синтеза текстур

Построив график  $\xi(k)$ , можно увидеть, как меняется плотность черных пикселей и прослеживается ли тренд.



Прохождение окном,  $W$ ,  $H$  - размеры изображения,  $w$  - ширина окна

# Оценка качества синтеза текстур






















В качестве метрики можно взять среднеквадратичную ошибку:

$$\xi = \frac{K}{W - w} \sum_{k=1}^{W-w} (\xi_k - \xi_{0k})^2,$$

где  $\xi_{0k}$  - это  $\xi_k$ , усредненное по изображениям, содержащим истинный тренд, а  $K$  - нормировочный множитель, вводимый для того, чтобы метрики сетей, обученных на разных выборках можно было сравнивать между собой.

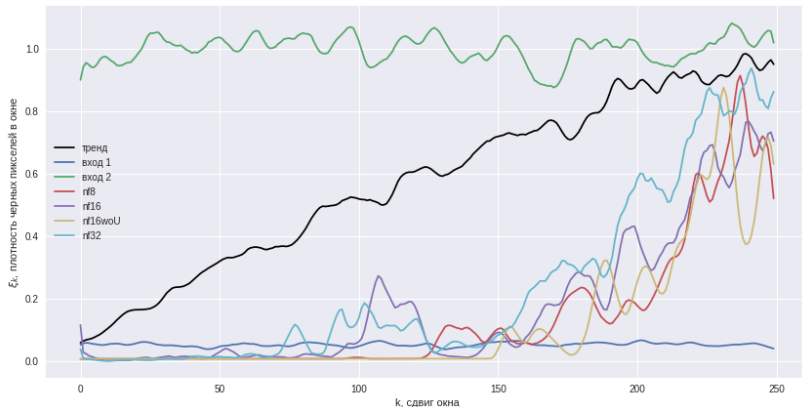
# Результаты синтеза

Выборка 1: 3000 обучающих троек, 50 тестовых

Вход 1	Вход 2	Тренд	nf8	nf16	nf16woU	nf32
						
						
						

Примеры синтеза (Выборка 1)

# Результаты синтеза



Аппроксимация тренда различными сетями (Выборка 1)






















# Результаты синтеза

Сеть	Число фильтров на 1-ом слое	Метрика
nf16woU	16	0.24048
nf8	8	0.22511
nf16	16	0.18844
nf32	32	0.14589

Значения метрики для разных сетей (меньше - лучше)

# Результаты синтеза

Выборка 2: 6000 обучающих троек, 50 тестовых

Вход 1	Вход 2	Тренд	nf32e5	nf64e1	nf64e5	nf64e10
						
						
						

Примеры синтеза (Выборка 2)



# Результаты синтеза



Аппроксимация тренда различными сетями (Выборка 2)

# Результаты синтеза

Сеть	Число фильтров на 1-ом слое	Метрика
nf64e10	64	0.11168
nf64e5	64	0.06501
nf32e5	32	0.04827
nf64e1	64	0.01393

Значения метрики для разных сетей (меньше - лучше)

# Выводы

- ▶ Предложен и исследован метод синтеза текстур с трендами
- ▶ Исследовано применение архитектуры GAN для синтеза текстур с трендами
- ▶ Получены результаты синтеза при некоторых наборах гиперпараметров сети на нескольких выборках
- ▶ Проведено измерение качества генерации для каждого из наборов

Результаты показывают на возможность применения GAN для синтеза текстур с трендами.

Спасибо за внимание!

# Задача минимизации

Обучение нейронной сети является задачей многопараметрической минимизации функционала потерь. Для используемых в этой работе сетей данная задача ставится так:

$$L(G, D) = L_{adv}(G, D) + \eta L1$$

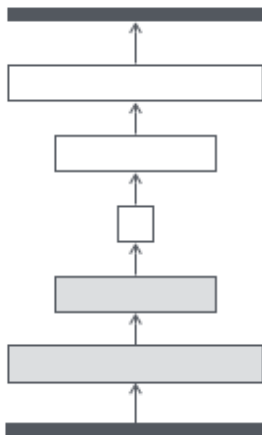
$$L1 = \mathbb{E}_{p_{data}(s_1, s_2, r)}(\| r - G(s_1, s_2) \|_1)$$

$$L_{adv}(G, D) = \mathbb{E}_{p_{data}(s_1, s_2, r)} \log D(s_1, s_2, r) + \\ + \mathbb{E}_{p_{data}(s_1, s_2)} \log(1 - D(s_1, s_2, G(s_1, s_2)))$$

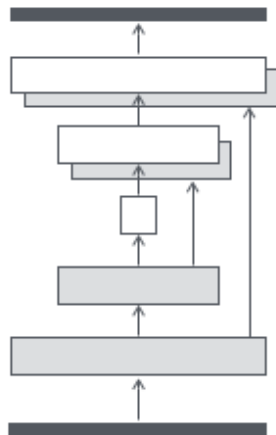
$$D^* = \arg \min_D L(G^*, D)$$

$$G^* = \arg \min_G L(G, D^*)$$

# Архитектуры G и D



Encoder-decoder



U-Net

Схематическое изображение нейросети-генератора

# Архитектуры G и D

Генератор:

C[nf]-C[nf\*2]-C[nf\*4]-C[nf\*8]-C[nf\*8]-C[nf\*8]-C[nf\*8]-C[nf\*8]-  
DC[nf\*8]-DC[nf\*8]-DC[nf\*8]-DC[nf\*8]-DC[nf\*4]-DC[nf\*2]-  
DC[nf]-DC[1]

Под C[nf] или DC[nf] здесь подразумеваются блоки, состоящие из сверточного или разверточного слоя с указанным числом фильтров, батч-нормализации и функции активации LeakyReLU с коэффициентом 0.2.

Дискриминатор:

C[nf]-C[nf\*2]-C[nf\*4]-C[nf\*8]-C[1]

В дискриминаторе батч-нормализация не применялась.