

Обзор статьи: Saad, F., Burnim, J., Carroll, C. *et al.* Scalable spatiotemporal prediction with Bayesian neural fields. *Nat Commun* 15, 7942 (2024). <https://doi.org/10.1038/s41467-024-51477-5>

Масштабируемое пространство-временное предсказание с байесовскими нейронными полями.

Пространственно-временные данные — это данные, которые состоят из измерений, собранных в разное время и в разных местах:

- метеорология (прогноз объема осадков или скорости ветра)
- эпидемиология (прогнозирование активных случаев гриппа) и
- городское планирование (предсказание перегруженности пассажиров на станциях метро)

В отличие от традиционных задач машинного обучения, где данные распределены независимо и одинаково, пространственно-временные данные требуют обработки своей нестационарной динамики как во временной, так и в пространственной областях. Кроме того, такие данные обусловлены неизвестными и шумно наблюдаемыми процессами. Следовательно, требуют учета алеаторической (неопределенность, связанная со случайностью в наблюдаемом процессе) и эпистемологической (неопределенность, связанная с недостатком знаний о системе) неопределенности в данных.

Доминирующий подход к моделированию пространно-временных данных основан на гауссовых процессах. Это вероятностная модель, которая моделирует распределение функций и затем аппроксимирует их. Она является гибкой и способна обрабатывать отсутствующие данные, но имеет слишком большую вычислительную сложность ($O(N^3)$, где N — это количество наблюдений => неприменима к большим выборкам) и требует подробных знаний об области применения.

Итак, во избежание проблем с гауссовскими процессами, авторы предлагают рассмотреть модель BAYESNF (*Bayesian Neural Field*) - Байесовское нейронное поле. Это вероятностная модель, которая объединяет концепции нейронных полей и байесовской статистики.

Нейронные поля (NF): это тип нейронной сети, предназначенный для работы с непрерывными данными, такими как изображения, сигналы или пространственные данные. Они могут представлять функции в непрерывном пространстве и обладают свойствами, которые делают их подходящими для моделирования физических явлений или интерполяции данных.

Байесовская статистика: это подход к статистике, основанный на теореме Байеса. Вместо того чтобы искать единственное "наилучшее" значение для параметров модели (как в классическом машинном обучении),

байесовский подход рассматривает параметры как случайные величины с определенным распределением вероятностей. Это позволяет учитывать неопределенность в данных и в самой модели.

Следовательно, BNF берет архитектуру нейронного поля и применяет к ней байесовские методы. Это означает, что вместо фиксированных весов и смещений, как в обычной нейронной сети, BNF имеет распределения вероятностей над этими параметрами. Это позволяет модели: оценивать неопределенность (BNF может предсказывать не только значение функции, но и вероятность того, что это значение верно), обучаться на малом количестве данных и адаптироваться к новым данным.

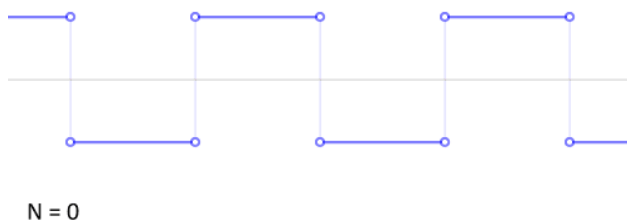
Чтобы подробнее разобраться в работе BAYESNF дадим несколько определений:

- 1) Априорное распределение (Prior Distribution) — это первоначальное убеждение о вероятности события до получения каких-либо новых данных.
- 2) Правдоподобие (Likelihood) — это вероятность наблюдать полученные данные, предполагая, что определенное событие произошло. Другими словами, насколько хорошо данные соответствуют определенной гипотезе.
- 3) Апостериорное распределение (Posterior Distribution) — это обновленное убеждение о вероятности события после учета новой информации (данных). Оно получается путем комбинирования априорного распределения и правдоподобия с помощью теоремы Байеса.

Пример (для понимания).

- **Априорное распределение:** Вероятность наличия этой болезни в общей популяции (например, 1%).
- **Данные:** Результаты медицинского теста.
- **Правдоподобие:** Вероятность получить такие результаты теста, если у пациента есть болезнь (например, 90%), и вероятность получить такие результаты, если болезни нет (например, 10%).
- **Апостериорное распределение:** Обновленная вероятность наличия болезни у пациента после получения результатов теста. Она будет выше, чем априорная вероятность, если тест положительный, и ниже, если тест отрицательный.

BAYESNF начинает с присваивания параметрам априорного распределения.



Далее, как и в гауссовских процессах, создает апостериорное распределение, учитывая наблюдаемые данные. Также чтобы учитывать разные частоты в данных (например, медленные сезонные изменения погоды и кратковременные дожди)

добавляется функция Фурье к необработанным данным времени и местоположения.

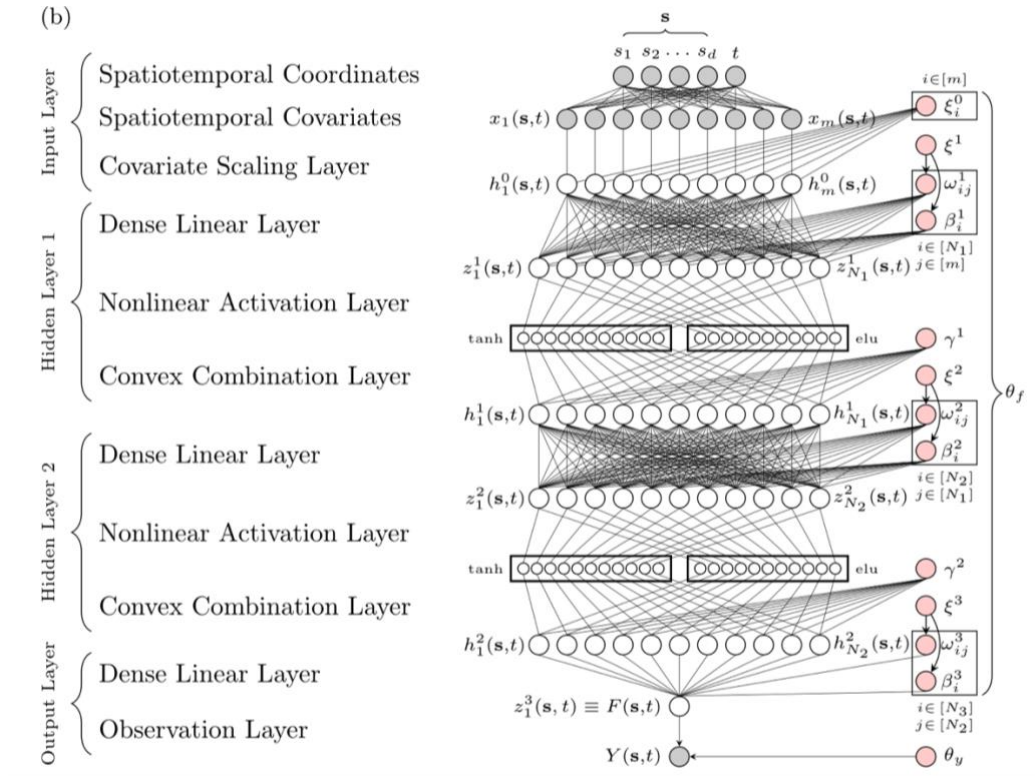
**Функции Фурье — это мощный математический инструмент, позволяющий разложить сложный сигнал на более простые составляющие — синусоиды и косинусоиды.*

Включение синусоидальных особенностей сезонности позволяет моделям BAYESNF также делать прогнозы на основе (многочисленных) сезонных эффектов. Все эти характеристики обеспечивают современную производительность с точки зрения точечных прогнозов и 95% интервалов прогнозирования на различных пространственно-временных наборах данных, без необходимости серьезной настройки структур модели BAYESNF на основе каждого набора данных. BAYESNF определяется по непрерывным пространственно-временным координатам, что позволяет прогнозировать в новых местах и в произвольные моменты времени.

Описание модели.

1.

- Набор данных $D = \{y(s_i, t_i) \mid i = 1, \dots, N\}$ N пространственно-временных наблюдений, где: s_i обозначает d -мерную пространственную координату, а t_i — временной индекс.
- $Y(s, t)$ - наблюдаемые данные
- $F(s, t)$ - скрытое, ненаблюдаемое поле. Оно представляет собой "истинное" значение измеряемой величины, без шума и ошибок наблюдения. Модель стремится оценить F на основе наблюдаемых данных Y , так как мы не можем напрямую измерить ее значение
- $x(s, t)$ — набор ковариат (независимые переменные, влияющие на F , например, высота над уровнем моря влияет на атмосферное давление)
- Θ_y — это параметры, которые управляют тем, как наблюдаемые данные $Y(s, t)$ связаны со скрытым полем $F(s, t)$. Они описывают шум, вариативность и другие аспекты процесса наблюдения
- Θ_f — это параметры, определяющие, как скрытое поле $F(s, t)$ генерируется в зависимости от ковариат $x(s, t)$



Авторы описывают поле, используя иерархическую байесовскую модель, состоящую из трех уровней: наблюдательной модели, модели процесса и параметрической модели.

$$\text{Observation Model: } [Y(\cdot) | F(\cdot), \Theta_y], \quad (2)$$

$$\text{Process Model: } [F(\cdot) | x(\cdot), \Theta_f], \quad (3)$$

$$\text{Parameter \setminus Models: } [\Theta_y, \Theta_f]. \quad (4)$$

- **Observation Model:** $[Y(\cdot) | F(\cdot), \Theta_y]$ — Это модель наблюдений. Она описывает, как наблюдаемые данные $Y(s, t)$ зависят от скрытого пространственно-временного поля $F(s, t)$ и параметров Θ_y .
- **Process Model:** $[F(\cdot) | x(\cdot), \Theta_f]$ — Это модель процесса. Она описывает, как скрытое поле $F(s, t)$ генерируется. F зависит от набора ковариат $x(s, t)$ и параметров Θ_f .
- **Parameter Models:** $[\Theta_y, \Theta_f]$ — Это модели параметров. Они описывают априорные распределения для параметров Θ_y и Θ_f .

2. Ковариаты — это независимые переменные, которые используются для объяснения или предсказания значений скрытого поля. Ковариаты могут включать следующие функции (т. е. модель может использовать различные математические функции от пространственно-временных координат (s, t) в качестве входных данных, чтобы лучше предсказывать F):

- *LinearTerms*: Прямое использование координат (t, s_i) как признаков. Это говорит о том, что модель предполагает линейную зависимость скрытого поля от времени и пространственных координат.

- *Взаимодействия (Temporal-Spatial Interactions, Spatial-Spatial Interactions)*: Произведения координат $(ts_i, s_i s_j)$. Это позволяет модели учесть *нелинейные* взаимодействия между временем и пространством, или между различными пространственными измерениями. Например, влияние времени может быть сильнее в одних областях пространства, чем в других.

- *Фурье-признаки (Spatial Fourier Features)*: Результаты преобразования Фурье координат. Это позволяет модели захватывать *периодические* паттерны во времени (сезонность) и в пространстве (пространственные периодичности). Они позволяют модели эффективно улавливать как низкочастотные (медленные изменения), так и высокочастотные (быстрые изменения) вариации.

3. Ковариантный масштабирующий слой

Каждый из m ковариатов $x_i(s, t)$ масштабируется случайным коэффициентом α_i . Эти масштабирующие коэффициенты α_i рассматриваются как случайные величины, распределенные логарифмически-нормально. Таким образом, устраняется необходимость ручной настройки масштабирования.

4. Скрытые слои

Модель содержит $L+1$ скрытых слоев, каждый слой (от 1 до L) содержит N^ℓ нейронов (единиц).

$$(\xi^\ell, \gamma_1^\ell, \dots, \gamma_{A^\ell}^\ell) \sim \text{iid Normal}(0, 1), \quad z_i^\ell(\mathbf{s}, t) := \sum_{j=1}^{N^{\ell-1}} \frac{\omega_{ij}^\ell}{\sqrt{N^{\ell-1}}} h_j^{\ell-1}(\mathbf{s}, t) + \beta_i^\ell$$

$$(\omega_{i1}^\ell, \dots, \omega_{iN^{\ell-1}}^\ell, \beta_i^\ell) \sim \text{iid Normal}(0, \sigma^\ell) \quad h_i^\ell(\mathbf{s}, t) := \sum_{j=1}^{A^\ell} \frac{e^{y_j^\ell}}{\sum_{k=1}^{A^\ell} e^{y_k^\ell}} u_j^\ell(z_i^\ell(\mathbf{s}, t))$$

$$\text{where } \sigma^\ell := \ln(1 + e^{\xi^\ell}) \quad (\text{only if } \ell < L + 1)$$

Выход каждого нейрона в слое l сначала вычисляется по формуле $z^\ell(s, t)$. Веса и смещения являются случайными величинами, распределенными независимо и одинаково согласно нормальному распределению $N(0, \sigma^{\ell 2})$. Дисперсия $\sigma^{\ell 2}$ является обучаемым параметром, его априорное распределение получается применением функции *softplus* к случайной величине $\xi^\ell \sim N(0, 1)$. $\text{softplus}(x) = \log(1 + \exp(x))$ — это функция, которая гарантирует, что дисперсия всегда положительна.

Вместо одной функции активации, каждый скрытый слой l использует A^ℓ разных функций активации (например, \tanh и elu). Выход нейрона — это взвешенная сумма результатов применения всех A^ℓ функций активации к z_i^ℓ . Используя обучаемые выпуклые комбинации "базовых" функций активации, модель автоматически определяет наиболее подходящую ковариантную

структуру для скрытого поля, что является важным преимуществом данного подхода.

Скрытое поле $F(s, t)$ определяется как выход последнего $(L+1)$ слоя, который содержит только один нейрон.

$$F(\mathbf{s}, t) := z_1^{L+1}(\mathbf{s}, t)$$

$F(s, t)$ - выход последнего слоя сети.

5. Наблюдательный слой

$$Y(\mathbf{s}, t) \sim \text{Dist}(F(\mathbf{s}, t), \Theta_y)$$

Наблюдательный слой служит для объединения $F(s, t)$, генерируемого последним слоем сети, с фактическими наблюдаемыми данными $Y(s, t)$. Он делает это посредством модели шума. Модель шума захватывает алеаторическую неопределенность в данных (неопределенность, связанная со случайностью в наблюдаемом процессе). Модель шума параметризуется вектором параметров Θ_y . Для параметров Θ_y существует априорное распределение μ_y .

$Y(s, t)$ связан с $F(s, t)$ через некоторое распределение вероятностей Dist .

В зависимости от типа данных $Y(s, t)$, могут использоваться разные распределения наблюдений. Вот несколько примеров, представленных в тексте: гауссовское, распределение Стьюдента и Пуассона.

6. Апостериорный вывод

Точно вычислить апостериорное распределение аналитически невозможно. Это связано со сложностью модели BAYESNF (глубокая нейронная сеть) и нелинейностью связей между параметрами и данными. Следовательно, необходимо прибегнуть к приближенным методам:

- Максимальные апостериорные ансамбли (Maximum a-posteriori ensembles - MAP)

Этот метод получает одно фиксированное значение весов, соответствующее максимуму апостериорной вероятности

- Вариационные ансамбли вывода (Variational inference ensembles - VI).

Этот метод получает аппроксимацию апостериорного распределения (те его приближение)

1) Выбирается форма $q(\theta)$ – более простое распределение: например, гауссовское распределение с параметрами μ, σ .

2) Определяется ELBO (Evidence Lower Bound) — функция, которая максимизируется для нахождения минимального расхождения между $q(\theta)$ и настоящим апостериором

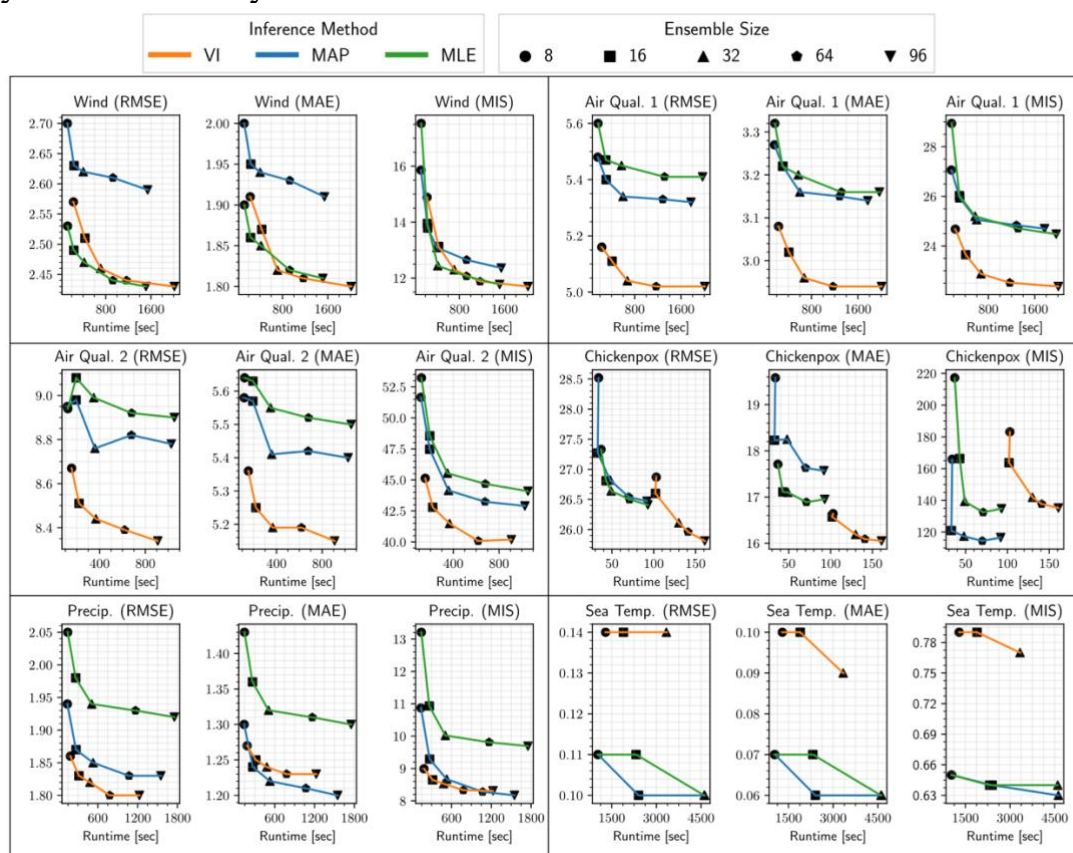
$$\text{ELBO} = \mathbb{E}_{q(\theta)}[\log p(\mathcal{D} | \theta)] - \text{KL}(q(\theta) \parallel p(\theta))$$

3) Оптимизация параметров $q(\theta)$ (например, μ, σ) с помощью стохастического градиентного спуска. => получение аппроксимации апостериорного распределения

- Метод максимального правдоподобия (Maximum Likelihood Estimation - MLE),

Он игнорирует априорную информацию, то есть не учитывает неопределённость в параметрах, просто находит такие значения параметров модели, которые максимально вероятно объясняют данные.

На рисунке показано сравнение профилей времени выполнения и точности с использованием трех методов вывода параметров для BAYESNF-VI, MAP и MLE. MLE работает не лучше, чем MAP или VI во всех профилях (и, как правило, хуже), иллюстрируя преимущества априорных параметров и апостериорной неопределенности. Между MAP и VI последний работает лучше в 13/18 случаях.



*Ensemble size (размер ансамбля) — это количество параметров одной модели

Далее, чтобы оценить эффективность BAYESNF авторы сравнивают результаты этой модели с 5-ю другими популярными моделями на 6 различных наборах данных.

BAYESNF с вариационным выводом (VI) — сильнейшая базовая модель. Ее метрика является лучшей на 4/6 наборах данных (скорость ветра, качество воздуха в Германии и Лондоне, накопленные осадки). BAYESNF с использованием MAP немного уступает, но все же лучшая на данных о ежемесячных аномалиях температуры поверхности моря в Тихом океане. И всего в одном наборе данных (о распространении ветряной оспы в Венгрии)

ошибка BAYESNF немного выше, чем базовый уровень STGLMM, хотя время работы последнего ~ в 4 раза выше.

В итоге, авторы делают вывод, что BNF мощный и гибкий подход к моделированию сложных данных, учитывая неопределенность и позволяя делать более надежные предсказания. Они особенно полезны в ситуациях, когда данных мало или они шумные.

Обзор кода

Для построения модели BAYESNF используется 3 файла: `inference`, `models`, `spatiotemporal`. `Inference` содержит функции для перемешивания данных, расчёта квантилей для нормальных и смешанных распределений, построения прогноза и алгоритмы обучения (MAP, MLE, VI). `Models` включает создание сезонных признаков, фурье-преобразования, а также построение модели `BayesianNeuralField1D` и её обучение. `Spatiotemporal` осуществляет предобработку временных данных, создание обработчика данных и реализацию моделей для временных и пространственно-временных задач.

В блоке `inference`:

- `permute_dataset` – перемешивает данные
- `_normal_quantile_via_root` - вычисляет q-квантиль среднего нормального распределения по заданным `means` и `scales` (нужно для байесовских нейронных сетей, где требуется интервальная оценка)
- `_approximate_normal_quantile` - приближённый способ найти квантили для смешанного распределения (вместо точного метода `_normal_quantile_via_root` делается приближение)
- `_get_percentile_normal` - вычисляет квантильные значения для смеси нормальных распределений, заданной по батчам средних (`means`) и стандартных отклонений (`scales`), по списку квантилей `quantiles`
- `_make_forecast_inner` - возвращает параметры распределения, необходимые для построения прогноза (среднее, дисперсия и пр.) — в зависимости от выбранного распределения
 - **NORMAL** (нормальное),
 - **NB** (негативное биномиальное),
 - **ZINB** (ZINB — Zero Inflated Negative Binomial) распределения.
- `forecast_parameters_batched` - выполняет батчевый прогноз параметров распределения. Она применяется, когда объём данных слишком большой для обработки за один проход, и позволяет: разбить входные

данные на батчи, прогнать их через модель `forecast_inner` и собрать параметры выходного распределения

- `make_vi_init` - создает инициализацию для вариационного вывода в виде случайных значений, которые затем будут оптимизированы
- `make_model` - создает модель `BayesianNeuralField1D` с заданными гиперпараметрами
- `make_prior` – используется для генерации априорных распределений модели
- `_build_observation_distribution` - возвращает объект распределения для `Negative Binomial` или `Zero-Inflated Negative Binomial` в зависимости от аргумента `distribution`
- `_get_nb_quantiles_root` - вычисление квантилей для негативного биномиального распределения с использованием метода нахождения корня
- `fit_vi` - обучение модели с использованием вариационного вывода(VI)
- `fit_map` - обучения модели с использованием метода максимума апостериорной вероятности (MAP)
- `_make_init_fn` - создает функцию инициализации параметров модели, которая использует априорное распределение `prior_d` и модифицирует его для более подходящей начальной инициализации параметров перед обучением
- `predict_bnf` - выполняет прогнозирование на новых данных с использованием уже обученной BNF-модели
- `ensemble_map` - реализует обучение ансамбля моделей методом MAP
- `ensemble_vi` – реализует ансамбль вариационных аппроксимаций (VI), используя аппроксимирующее апостериорное распределение и оптимизирует ELBO

В блоке `models`:

- `make_seasonal_frequencies` - вычисляет уникальные частоты для сезонных компонентов модели времени с использованием гармоник Фурье

- `make_seasonal_features` - создают сезонные признаки для заданных временных рядов, представляя сезонность через синусоиды и косинусоиды
- `make_fourier_features` - генерируют синусоиды и косинусоиды для разных частот, основанных на степенях, и создают набор признаков для дальнейшего использования в моделях
- `prior_model_fn` - создает априорные распределения для модели
- `make_likelihood_model` - создает модели правдоподобия для прогнозирования, в зависимости от типа распределения
- `BayesianNeuralField1D` - определяет байесовскую нейронную сеть, которая включает в себя различные преобразования признаков (признаки Фурье, сезонные признаки, условия взаимодействия)

В блоке `spatiotemporal`:

- `seasonality_to_float` - позволяет преобразовать строковое представление временных интервалов в число с плавающей точкой, которое показывает, сколько таких периодов (например, дней или недель) в одном "сезонном" периоде (например, год)
- `seasonalities_to_array` предназначена для преобразования списка сезонных значений (как чисел, так и строк) в массив с плавающими значениями, представляющими продолжительность, выраженную в единицах времени относительно частоты данных
- `_convert_datetime_col` - выполняет предобработку временной колонки таблицы, преобразуя значения времени в числовой формат, пригодный для обучения моделей
- `SpatiotemporalDataHandler` — универсальный обработчик данных для временных и пространственно-временных задач машинного обучения. Он включает нормализацию признаков, преобразование временных меток и подготовку выборок для обучения и тестирования
- `BayesianNeuralFieldEstimator` - каркас для создания вероятностных нейросетей над временными рядами, с возможностью учитывать сезонность, взаимодействия признаков, фурье-преобразования и разную структуру выходных распределений
- `BayesianNeuralFieldMAP` - ансамбль нейросетей, обученных по принципу MAP-оценки

- BayesianNeuralFieldMLE - ансамбль нейросетей, обученных по принципу MLE-оценки
- BayesianNeuralFieldVI - ансамбль нейросетей, обученных по принципу VI-оценки

Проверка модели на данных.

Мы запустили модель BAYESNF на данных о еженедельных случаях заболевания ветряной оспой в 20 округах Венгрии в период с 2005 по 2014 год. В тренировочной набор определили данные 2005–2012 гг., в тестовый – 2013–2014 гг.

Результаты:

	Jax	PyTorch
RMSE	32.79481506347656	35.50316541406665

`!pip install git+https://github.com/ArinaGri/DL_project.git` – ссылка для использования нашего пакета в ЮпитерЛаб

```
#!pip install git+https://github.com/ArinaGri/DL_project.git
```

```
import pandas as pd
import torch
df_train = pd.read_csv('train1.csv', parse_dates=['datetime'])
df_test = pd.read_csv('test1.csv', parse_dates=['datetime'])
```

```
import bayes

model = bayes.BayesianNeuralFieldMAP(width=256,
    depth=6,
    freq='W',
    seasonality_periods=['M', 'Y'], # equivalent to [365.25/12, 365.25]
    num_seasonal_harmonics=[2, 10], # two harmonics for M; one harmonic for Y
    feature_cols=['datetime', 'latitude', 'longitude'], # time, spatial 1, ..., spatial n
    target_col='chickenpox',
    observation_model='NB',
    timetype='index',
    #standardize=['latitude', 'longitude'],
    interactions=[(0, 1), (0, 2), (1, 2)])
```

Используется устройство: cpu

```
model = model.fit(
    df_train,
    seed=torch.manual_seed(0),
    ensemble_size=32,
    num_epochs=5,
)
```

```
Epoch [1/5], Loss: 3888.8886795043945
Epoch [2/5], Loss: 24399.392517089844
Epoch [3/5], Loss: 2612.8295974731445
Epoch [4/5], Loss: 14510.98796081543
Epoch [5/5], Loss: 5416.640586853027
```