SPATIO TEMPROTAL LONG-TERM AND SHORT-TERM FORECASTING

Чеснокова Алина
Гришаева Арина


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)

ПРОСТРАНСТВЕННО-ВРЕМЕННЫЕ ДАННЫЕ

1

требуют обработки своей нестационарной динамики как во временной, так и в пространственной областях

2

обусловлены неизвестными и шумно наблюдаемыми процессами

ГАУССОВСКИЕ ПРОЦЕССЫ

- Вероятностная модель
- Строит доверительные интервалы
- Моделирует распределение функций
- Аппроксимирует их

НО!

- имеет слишком большую вычислительную сложность $O(N3)$
- требует подробных знаний об области применения

BAYESNF

BAYESNF - 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.*

Байесовская статистика: *вместо того чтобы искать единственное "наилучшее" значение для параметров модели, байесовский подход рассматривает параметры как случайные величины с определенным распределением вероятностей.*

ВМЕСТО ФИКСИРОВАННЫХ ВЕСОВ И СМЕЩЕНИЙ, КАК В ОБЫЧНОЙ НЕЙРОННОЙ СЕТИ, BNF ИМЕЕТ РАСПРЕДЕЛЕНИЯ ВЕРОЯТНОСТЕЙ НАД ЭТИМИ ПАРАМЕТРАМИ. ЭТО ПОЗВОЛЯЕТ МОДЕЛИ:

- оценивать неопределенность
- обучаться на малом количестве данных
- адаптироваться к новым данным

BAYESNF

01

присваивает параметрам априорное распределение

02

создает апостериорное распределение, учитывая наблюдаемые данные

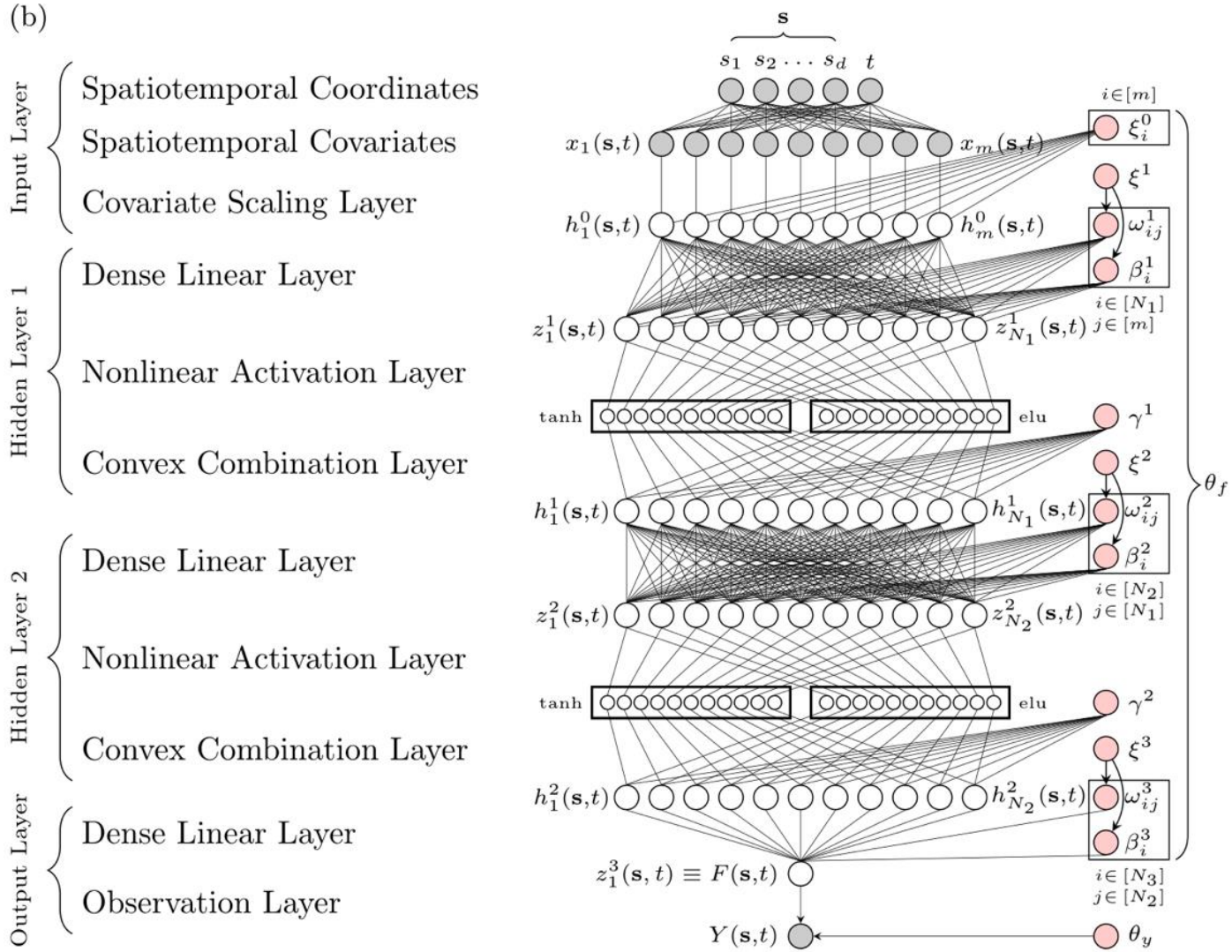
03

Для учета разной частоты в данных добавляется функция Фурье

ОПИСАНИЕ МОДЕЛИ.

- Набор данных D = {y (si, ti) i = 1, ..., N} N пространственно-временных наблюдений, где: si обозначает d-мерную пространственную координату, а ti – 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)
- Θy — 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
- Θf — это параметры, определяющие, как скрытое поле F (s, t) генерируется в зависимости от ковариат x (s, t)

(b)

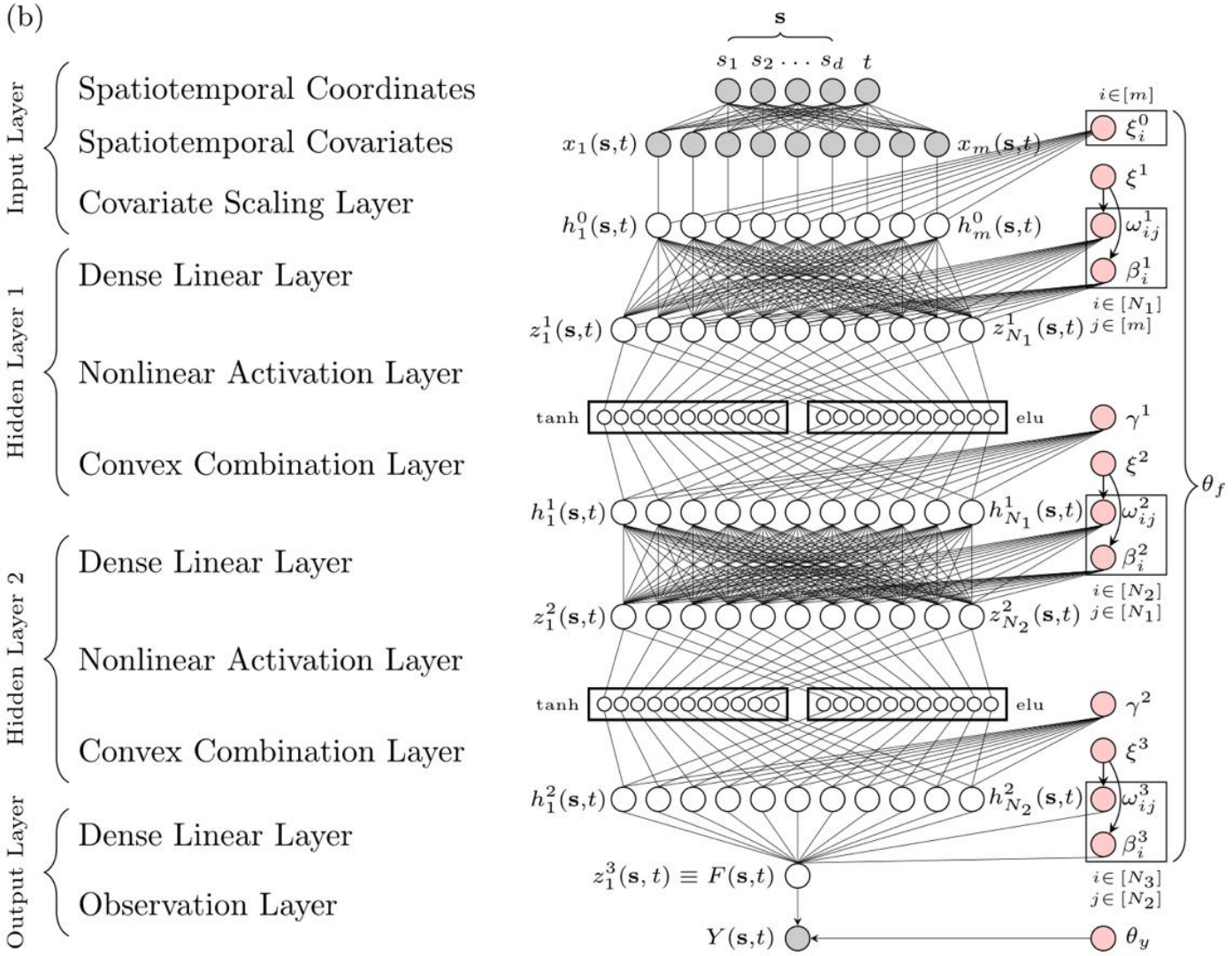

Input Layer
Spatiotemporal Coordinates
Spatiotemporal Covariates
Covariate Scaling Layer
Hidden Layer 1
Dense Linear Layer
Nonlinear Activation Layer
Convex Combination Layer
Hidden Layer 2
Dense Linear Layer
Nonlinear Activation Layer
Convex Combination Layer
Output Layer
Dense Linear Layer
Observation Layer
s
s_1 s_2 $\cdots$ s_d t
$x_1(\mathbf{s},t)$
$x_m(\mathbf{s},t)$
$h_1^0(\mathbf{s},t)$
$h_m^0(\mathbf{s},t)$
$z_1^1(\mathbf{s},t)$
$z_{N_1}^1(\mathbf{s},t)$
tanh
elu
$h_1^1(\mathbf{s},t)$
$h_{N_1}^1(\mathbf{s},t)$
$z_1^2(\mathbf{s},t)$
$z_{N_2}^2(\mathbf{s},t)$
tanh
elu
$h_1^2(\mathbf{s},t)$
$h_{N_2}^2(\mathbf{s},t)$
$z_1^3(\mathbf{s},t) \equiv F(\mathbf{s},t)$
$Y(\mathbf{s},t)$
$i \in [m]$
ξ_i^0
ξ^1
ω_{ij}^1
β_i^1
$i \in [N_1]$
$j \in [m]$
γ^1
ξ^2
ω_{ij}^2
β_i^2
$i \in [N_2]$
$j \in [N_1]$
γ^2
ξ^3
ω_{ij}^3
β_i^3
$i \in [N_3]$
$j \in [N_2]$
θ_f
θ_y

КОВАРИАТЫ МОГУТ ВКЛЮЧАТЬ СЛЕДУЮЩИЕ ФУНКЦИИ:

- *LinearTerms*

- *Взаимодействия (Temporal-Spatial Interactions, Spatial-Spatial Interactions)*

- *Фурье-признаки (Spatial Fourier Features)*



- Серые круги – наблюдаемые переменные
- Белые – локальные скрытые переменные
- Розовые – глобальные параметры, которые являются общими для всех пространственно-временных координат

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

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

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

АПОСТЕРИОРНЫЙ ВЫВОД

1

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

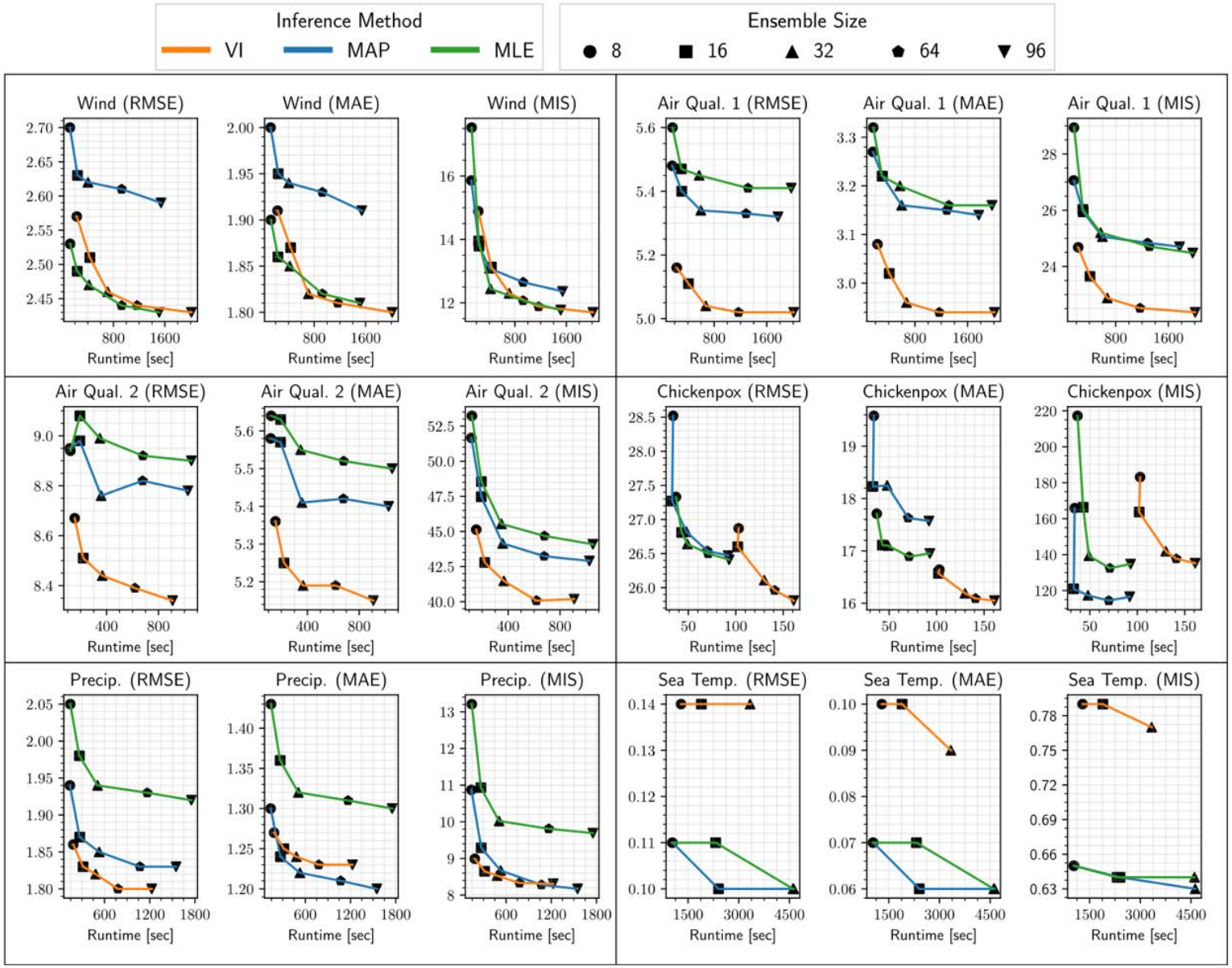
2

Вариационные ансамбли вывода (Variational inference ensembles - VI)

3

Метод максимального правдоподобия (Maximum Likelihood Estimation - MLE)

СРАВНЕНИЕ MAP, VI, MLE ПО ВРЕМЕНИ И МЕТРИКЕ



BAYESNF С ВАРИАЦИОННЫМ ВЫВОДОМ (VI)

		Prediction Error			
Dataset	**Method**	**RMSE**	**MAE**	**MIS**	**Runtime**
Wind Speed	Bayesian Neural Field (VI)	**2.44**	**1.81**	**11.88**	1167
	Bayesian Neural Field (MAP)	2.61	1.93	12.65	927
	Sparse Spatiotemporal Variational Gaussian Process	5.04	4.18	24.72	1112
	Spatiotemporal Gradient Boosting Trees	3.74	2.79	18.43	2907
	Neural Basis Expansion Analysis	5.20	4.07	22.92	9237
	Spatiotemporal Generalized Linear Mixed Model (All)	✗	✗	✗	✗
	Trend Surface Regression	4.94	3.88	24.83	≤1
Air Quality 1	Bayesian Neural Field (VI)	**5.02**	**2.94**	**22.52**	1169
	Bayesian Neural Field (MAP)	5.33	3.15	24.84	1284
	Sparse Spatiotemporal Variational Gaussian Process	6.24	3.91	35.59	1348
	Spatiotemporal Gradient Boosting Trees	7.42	4.40	31.56	5665
	Neural Basis Expansion Analysis	9.23	5.95	45.11	1461
	Spatiotemporal Generalized Linear Mixed Model (All)	✗	✗	✗	✗
	Trend Surface Regression	9.35	6.62	55.98	≤1
Air Quality 2	Bayesian Neural Field (VI)	**8.39**	**5.19**	**40.08**	618
	Bayesian Neural Field (MAP)	8.82	5.42	43.24	678
	Sparse Spatiotemporal Variational Gaussian Process	9.92	6.78	56.12	628
	Spatiotemporal Gradient Boosting Trees	8.77	5.57	43.71	2671
	Neural Basis Expansion Analysis	12.63	8.24	63.84	778
	Spatiotemporal Generalized Linear Mixed Model (AR1)	11.92	7.81	73.00	17100
	Spatiotemporal Generalized Linear Mixed Model (RW)	14.62	9.48	157.10	9447
	Spatiotemporal Generalized Linear Mixed Model (IID)	12.87	8.78	127.48	3545
	Trend Surface Regression	18.44	12.32	117.90	≤1
Precipitation	Bayesian Neural Field (VI)	**1.80**	**1.23**	**8.33**	778
	Bayesian Neural Field (MAP)	**1.83**	**1.21**	**8.28**	1069
	Sparse Spatiotemporal Variational Gaussian Process	3.14	2.27	31.00	1203
	Spatiotemporal Gradient Boosting Trees	2.63	1.67	11.13	2064
	Neural Basis Expansion Analysis	✗	✗	✗	✗
	Spatiotemporal Generalized Linear Mixed Model (All)	✗	✗	✗	✗
	Trend Surface Regression	3.61	2.69	20.81	≤1

BAYESNF С ИСПОЛЬЗОВАНИЕМ MAP

		Prediction Error			
Dataset	**Method**	**RMSE**	**MAE**	**MIS**	**Runtime**
Sea Surface Temperature	Bayesian Neural Field (VI)	0.14	0.09	0.77	3335
	Bayesian Neural Field (MAP)	**0.10**	**0.06**	**0.63**	4624
	Sparse Spatiotemporal Variational Gaussian Process	✗	✗	✗	✗
	Spatiotemporal Gradient Boosting Trees	0.45	0.33	1.94	12379
	Neural Basis Expansion Analysis	0.20	0.15	0.97	1120
	Spatiotemporal Generalized Linear Mixed Model (All)	✗	✗	✗	✗
	Trend Surface Regression	0.55	0.42	2.89	3

STGLMM

Dataset	Method	Prediction Error			Runtime
		RMSE	MAE	MIS	
Chickenpox Cases	Bayesian Neural Field (VI)	25.96	16.09	137.74	141
	Bayesian Neural Field (MAP)	26.54	17.63	**114.44**	70
	Sparse Spatiotemporal Variational Gaussian Process	32.00	21.22	212.87	63
	Spatiotemporal Gradient Boosting Trees	26.83	15.84	122.39	189
	Neural Basis Expansion Analysis	29.51	17.56	167.27	250
	Spatiotemporal Generalized Linear Mixed Model (AR1)	**25.30**	**15.26**	179.29	887
	Spatiotemporal Generalized Linear Mixed Model (RW)	26.92	16.79	179.63	386
	Spatiotemporal Generalized Linear Mixed Model (IID)	28.23	16.85	327.72	264
	Trend Surface Regression	29.75	21.30	172.43	≤1

КОД

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.

JAX → PYTORCH

```jax
def _normal_quantile_via_root(means, scales, q, axis=(0, 1)):
  n = tfd.Normal(means, scales)
  res = tfp.math.find_root_chandrupatla(
      lambda x: n.cdf(x).mean(axis) - q,
      low=jnp.amin(means) - 5 * jnp.amax(scales),
      high=jnp.amax(means) + 5 * jnp.amax(scales),
      value_tolerance=1e-5,
      max_iterations=60,
  )
  return res.estimated_root
```

```pytorch
def _normal_quantile_via_root(means: torch.Tensor, scales: torch.Tensor,
                              q: float, axis=(0, 1)) -> torch.Tensor:
    normal_dist = torch.distributions.Normal(means, scales)

    if isinstance(axis, int):
        valid_axes = (axis,) if axis < means.ndim else ()
    else:
        valid_axes = tuple(a for a in axis if a < means.ndim)

    def objective(x):
        cdf_vals = normal_dist.cdf(x)
        if valid_axes:
            return cdf_vals.mean(dim=valid_axes) - q
        else:
            return cdf_vals.mean() - q

    low = means.min() - 5 * scales.max()
    high = means.max() + 5 * scales.max()

    result = root_scalar(lambda x: objective(torch.tensor(x)),
                         bracket=[low.item(), high.item()], method='brentq')
    return torch.tensor(result.root)
```

JAX → PYTORCH

```jax
def make_seasonal_features(
    x: jax.typing.ArrayLike,
    seasonality_periods: np.ndarray,
    num_harmonics: np.ndarray,
    rescale: bool = False,
) -> jnp.ndarray:
  """Returns a set of cos and sin features for each seasonality
  x = jnp.reshape(x, (-1, 1))
  frequencies, harmonics = make_seasonal_frequencies(
      seasonality_periods, num_harmonics
  )
  y = 2 * jnp.pi * frequencies * x
  features = jnp.column_stack((jnp.cos(y), jnp.sin(y)))
  denominator = jnp.tile(harmonics, 2)
  return features / denominator if rescale else features
```

```pytorch
def make_seasonal_features(x: torch.Tensor, periods: np.ndarray,
                           harmonics: np.ndarray, rescale=False) -> torch.Tensor:
    x = x.view(-1, 1)
    freqs, harm = make_seasonal_frequencies(periods, harmonics)
    y = 2 * np.pi * torch.from_numpy(freqs).to(x.device) * x
    feats = torch.cat([torch.cos(y), torch.sin(y)], dim=1)
    if rescale:
        feats /= torch.from_numpy(np.tile(harm, 2)).to(x.device)
    return feats
```

ДАННЫЕ

- Данные о еженедельных случаях заболевания ветряной оспой в 20 округах Венгрии в период с 2005 по 2014 год.

	Jax	PyTorch
RMSE	32.79481506347656	35.50316541406665

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

СПАСИБО ЗА ВНИМАНИЕ!