Tensor Gaussian Process with Contraction for Tensor Regression

Hu Sun [1], Ward Manchester [2], Meng Jin[3], Yang Liu[4], Yang Chen [1]

[1]Department of Statistics, University of Michigan, Ann Arbor

[2]Climate and Space Sciences and Engineering (CLASP), University of Michigan, Ann Arbor

[3]Solar & Astrophysics Lab, Lockheed Martin

[4]W.W. Hansen Experimental Physics Lab, Stanford University

March 9, 2023

Scalar-on-Tensor Regression Problem

- Data: $\{\mathcal{X}_i, y_i\}_{i=1}^n$, where:
  - $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$: m-mode *tensor* covariates of size $I_1 \times I_2 \times \ldots \times I_m$.
  - $\{y_1, y_2, \ldots, y_n\}$: *scalar* regression labels.

Scalar-on-Tensor Regression Problem

- Data: $\{\mathcal{X}_i, y_i\}_{i=1}^n$, where:
  - $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$: m-mode *tensor* covariates of size $I_1 \times I_2 \times \ldots \times I_m$.
  - $\{y_1, y_2, \ldots, y_n\}$: *scalar* regression labels.
- Classic *Scalar-on-Tensor Regression* Model:

$$\mathbb{E}[y|\mathcal{X}] = \alpha + \langle \mathcal{W}, \mathcal{X} \rangle \tag{1}$$

where the regression coefficient $\mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_m}$ matches the size of the tensor covariates $\mathcal{X}$.

Scalar-on-Tensor Regression Problem

- Data: $\{\mathcal{X}_i, y_i\}_{i=1}^n$, where:
  - $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$: m-mode *tensor* covariates of size $I_1 \times I_2 \times \ldots \times I_m$.
  - $\{y_1, y_2, \ldots, y_n\}$: *scalar* regression labels.
- Classic *Scalar-on-Tensor Regression* Model:

$$\mathbb{E}[y|\mathcal{X}] = \alpha + \langle \mathcal{W}, \mathcal{X} \rangle \tag{1}$$

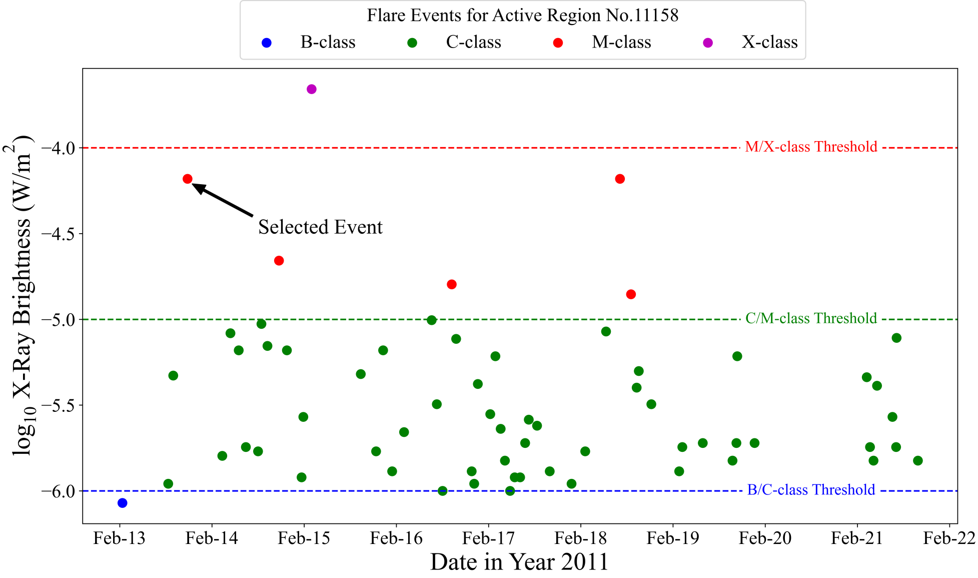
where the regression coefficient $\mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_m}$ matches the size of the tensor covariates $\mathcal{X}$.

- Model Dimensionality p:

$$\mathbb{E}[y|\mathcal{X}] = \alpha + \operatorname{vec}(\mathcal{X})^\top \underbrace{\operatorname{vec}(\mathcal{W})}_{p=\prod_{j=1}^m I_j} \tag{2}$$

and the dimensionality p increases very quickly as the tensor size grows in *any* mode.

An Astrophysics Example of Scalar-on-Tensor Regression



An Astrophysics Example of Scalar-on-Tensor Regression

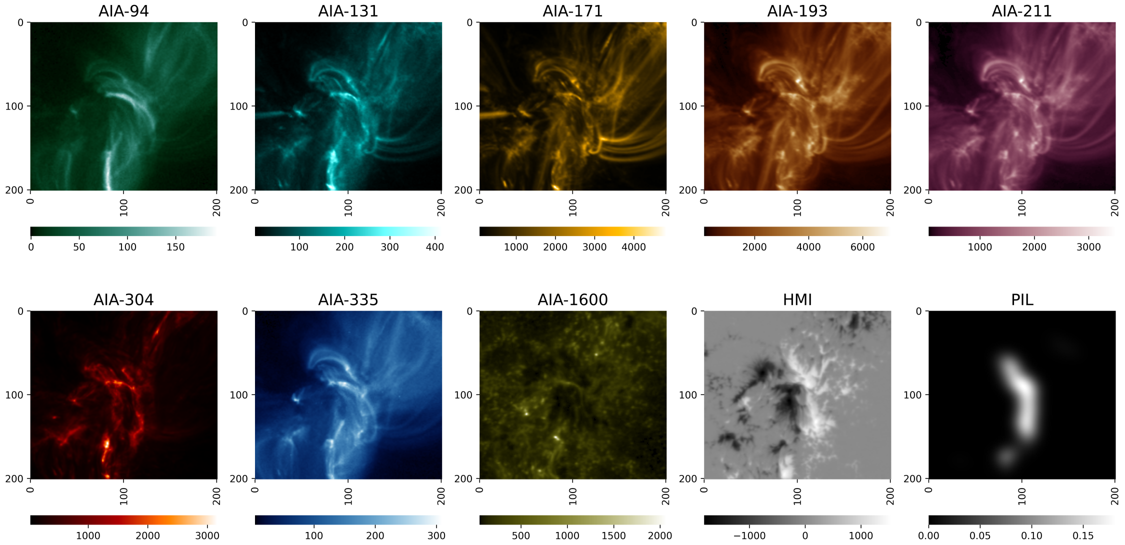


Figure: Tensor Data (size $= 201 \times 201 \times 10$) of the Selected Event.

Low-Rankness Assumption of $\mathcal{W}$

- Previous works (e.g. [1], [2]) propose to reduce the dimensionality of $\mathcal{W}$, i.e. the regression coefficient tensor, via a low-rankness assumption:

Low-Rankness Assumption of $\mathcal{W}$

- Previous works (e.g. [1], [2]) propose to reduce the dimensionality of $\mathcal{W}$, i.e. the regression coefficient tensor, via a low-rankness assumption:
  - CP-decomposition:

$$\mathcal{W} = \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \boldsymbol{\beta}_2^{(r)} \circ \ldots \circ \boldsymbol{\beta}_m^{(r)}$$

    where R is the rank of the tensor, and $\circ$ is vector outer product.

Low-Rankness Assumption of $\mathcal{W}$

- Previous works (e.g. [1], [2]) propose to reduce the dimensionality of $\mathcal{W}$, i.e. the regression coefficient tensor, via a low-rankness assumption:
  - CP-decomposition:
$$\mathcal{W} = \sum_{r=1}^{R} \boldsymbol{\beta}_1^{(r)} \circ \boldsymbol{\beta}_2^{(r)} \circ \ldots \circ \boldsymbol{\beta}_m^{(r)}$$
    where R is the rank of the tensor, and $\circ$ is vector outer product.
  - Tucker-decomposition:
$$\mathcal{W} = \mathcal{S} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \ldots \times_m \mathbf{U}_m^\top$$
    where $\mathcal{S}$ is a "core" tensor of size $I_1^{'} \times I_2^{'} \times \ldots \times I_m^{'}$, where $I_j^{'} << I_j$, and $\times_j$ is the j^{th}-mode product. $\mathbf{U}_j$ is an $I_j^{'} \times I_j$ orthogonal matrix with $\mathbf{U}_j \mathbf{U}_j^\top = \mathbf{I}_{I_j^{'}}$.

Tensor Gaussian Process Regression

- Given the Tucker Decomposition on $\mathcal{W}$:

$$\mathcal{W} = \mathcal{S} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \ldots \times_m \mathbf{U}_m^\top$$

- Further assume that $\mathcal{S}$ has a Gaussian prior:

$$\operatorname{vec}(\mathcal{S}) \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}_{d'}\right), \quad d' = \prod_{j=1}^{m} I_j' \tag{3}$$

- Then for any pair of tensor data $(\mathcal{X}_1, \mathcal{X}_2)$:

$$\operatorname{Cov}\left[\langle \mathcal{W}, \mathcal{X}_1 \rangle, \langle \mathcal{W}, \mathcal{X}_2 \rangle\right] = \operatorname{vec}(\mathcal{X}_1)^\top \left(\mathbf{U}_m^\top \mathbf{U}_m \otimes \mathbf{U}_{m-1}^\top \mathbf{U}_{m-1} \otimes \ldots \otimes \mathbf{U}_1^\top \mathbf{U}_1\right) \operatorname{vec}(\mathcal{X}_2)$$

where $\otimes$ is the matrix Kronecker product.

Tensor Gaussian Process Regression

- Tensor Gaussian Process (**Tensor-GP**) [3] is defined as:

$$
\begin{aligned}
y &= f(\mathcal{X}) + \epsilon && \text{(Likelihood)} \\
f(.) &\sim \mathbf{GP}\left(0, K(.,.)\right) && \text{(Gaussian Process Prior)} \\
K(\mathcal{X}_1, \mathcal{X}_2) &= \text{vec}\left(\mathcal{X}_1\right)^\top \left[\otimes_{j=1}^{m} \mathbf{K}_{m-j}\right] \text{vec}\left(\mathcal{X}_2\right) && \text{(Multi-Linear Kernel)} \\
\epsilon &\overset{i.i.d.}{\sim} \mathcal{N}\left(0, \sigma^2\right) && \text{(Additive Noise)}
\end{aligned}
$$

with kernel hyperparameters in red, and each $\mathbf{K}_{m-j}$ is low-ranked with $\mathbf{K}_{m-j} = \mathbf{U}_{m-j}{}^\top \mathbf{U}_{m-j}$.

Overview of Our Work

In this work:

- we consider a special type of 3-mode tensor: **multi-channel image**.
  - $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, with H: height, W: width, C: channel (modality).

Overview of Our Work

In this work:

- we consider a special type of 3-mode tensor: **multi-channel image**.
  - $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, with H: height, W: width, C: channel (modality).
- we model the scalar-on-tensor regression problem in two successive steps:

Overview of Our Work

In this work:

- we consider a special type of 3-mode tensor: **multi-channel image**.
  - $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, with H: height, W: width, C: channel (modality).
- we model the scalar-on-tensor regression problem in two successive steps:
  1. (*tensor contraction*) For each $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, we estimate a latent tensor $\mathcal{Z} \in \mathbb{R}^{h \times w \times C}$ via:
  $$\mathcal{Z} = h(\mathcal{X}) = \mathcal{X} \times_1 \mathbf{A}_{h \times H} \times_2 \mathbf{B}_{w \times W} \times_3 \mathbf{I}_C \tag{3}$$
  and $h << H, w << W$.

Overview of Our Work

In this work:

- we consider a special type of 3-mode tensor: **multi-channel image**.
  - $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, with H: height, W: width, C: channel (modality).
- we model the scalar-on-tensor regression problem in two successive steps:
  1. (*tensor contraction*) For each $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, we estimate a latent tensor $\mathcal{Z} \in \mathbb{R}^{h \times w \times C}$ via:
  $$\mathcal{Z} = h(\mathcal{X}) = \mathcal{X} \times_1 \mathbf{A}_{h \times H} \times_2 \mathbf{B}_{w \times W} \times_3 \mathbf{I}_C \tag{3}$$
  and $h << H, w << W$.
  2. (*tensor regression*) We model the GP regression problem over the set of latent tensors:
  $$y = g \circ h(\mathcal{X}) + \epsilon, \quad g(.) \sim \mathbf{GP}\left(0, K(.,.)\right)$$
  $$K(\mathcal{Z}_1, \mathcal{Z}_2) = \text{vec}\left(\mathcal{Z}_1\right)^\top \left(\mathbf{K}_3 \otimes \mathbf{K}_2 \otimes \mathbf{K}_1\right) \text{vec}\left(\mathcal{Z}_2\right)$$

Tensor Contraction

- Tensor contraction is an operation that **reduces the size** of an input tensor data while **keeping the tensor format** of the data.

Tensor Contraction

- Tensor contraction is an operation that **reduces the size** of an input tensor data while **keeping the tensor format** of the data.
- For multi-channel tensor $\mathcal{X}$, tensor contraction is conducted via:

$$\mathcal{Z} = \mathcal{X} \times_1 \mathbf{A}_{h \times H} \times_2 \mathbf{B}_{w \times W} \times_3 \mathbf{I}_C$$

  and tensor shape is reduced from $(H \times W \times C)$ to $(h \times w \times C)$.

Tensor Contraction

- Tensor contraction is an operation that **reduces the size** of an input tensor data while **keeping the tensor format** of the data.
- For multi-channel tensor $\mathcal{X}$, tensor contraction is conducted via:

$$\mathcal{Z} = \mathcal{X} \times_1 \mathbf{A}_{h\times H} \times_2 \mathbf{B}_{w\times W} \times_3 \mathbf{I}_C$$

  and tensor shape is reduced from $(H \times W \times C)$ to $(h \times w \times C)$.
- For each channel $c \in [C] \coloneqq \{1, 2, \dots, C\}$:

$$\mathcal{Z}^{(c)} = \mathbf{A}\mathcal{X}^{(c)}\mathbf{B}^\top$$

  where $\mathcal{Z}^{(c)}, \mathcal{X}^{(c)}$ are the c-th channel of $\mathcal{Z}$ and $\mathcal{X}$.

Tensor Contraction

- Tensor contraction is an operation that **reduces the size** of an input tensor data while **keeping the tensor format** of the data.
- For multi-channel tensor $\mathcal{X}$, tensor contraction is conducted via:

$$\mathcal{Z} = \mathcal{X} \times_1 \mathbf{A}_{h\times H} \times_2 \mathbf{B}_{w\times W} \times_3 \mathbf{I}_C$$

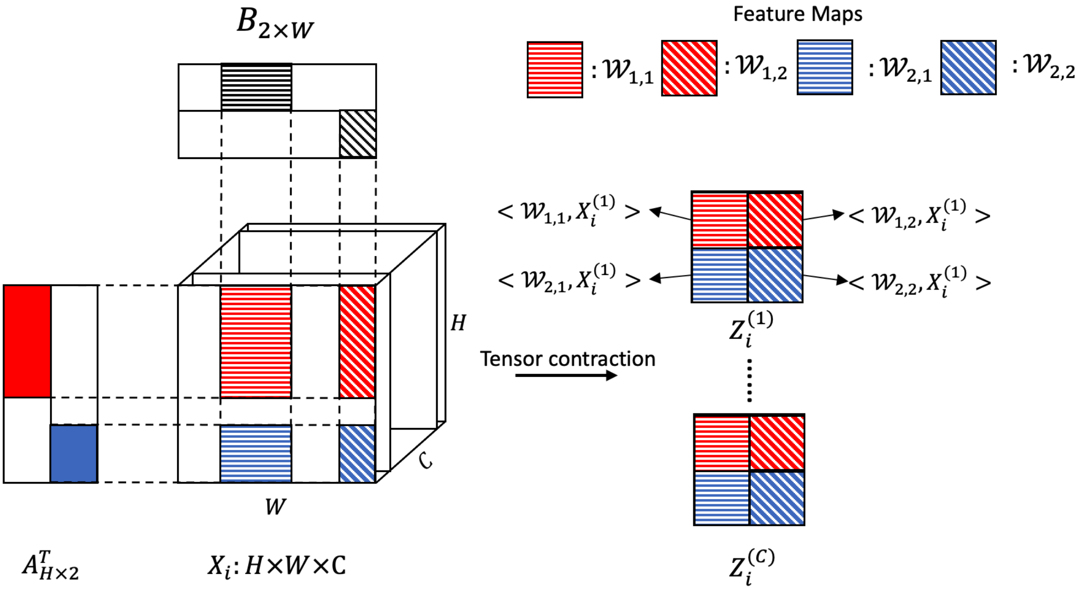
  and tensor shape is reduced from $(H \times W \times C)$ to $(h \times w \times C)$.
- For each channel $c \in [C] := \{1, 2, \ldots, C\}$:

$$\mathcal{Z}^{(c)} = \mathbf{A}\mathcal{X}^{(c)}\mathbf{B}^\top$$

  where $\mathcal{Z}^{(c)}, \mathcal{X}^{(c)}$ are the c-th channel of $\mathcal{Z}$ and $\mathcal{X}$.
- For each (s, t)-th element of $\mathcal{Z}^{(c)}$:

$$\mathcal{Z}^{(c)}(s,t) = \mathbf{A}(s,:)\mathcal{X}^{(c)}\left[\mathbf{B}(t,:)\right]^\top = \left\langle \underbrace{\left[\mathbf{A}(s,:)\right]^\top\left[\mathbf{B}(t,:)\right]}_{\text{rank-1 feature map } \mathbf{w}_{st}}, \mathcal{X}^{(c)} \right\rangle$$

Example of Tensor Contraction



Example of Tensor Contraction

- In the data tensor $\mathcal{X}$, pixels of $\mathcal{X}^{(c)}$ on the s^{th} row or t^{th} column share the same *spatial coordinates*.
- In the latent tensor $\mathcal{Z}$, pixels of $\mathcal{Z}^{(c)}$ on the s^{th} row or t^{th} column share the same *feature map basis vector* in $\mathbf{A}$ or $\mathbf{B}$.

Interpretable Tensor Contraction with Total-Variation Regularization

- Typically, we want to extract features from imaging data from **spatially-contiguous** regions.

Interpretable Tensor Contraction with Total-Variation Regularization

- Typically, we want to extract features from imaging data from **spatially-contiguous** regions.
- In tensor contraction, recall:

$$\mathcal{Z}^{(c)}(s,t) = \mathbf{A}(s,:)\mathcal{X}^{(c)}\left[\mathbf{B}(t,:)\right]^{\top} = \left\langle \mathbf{W}_{st}, \mathcal{X}^{(c)} \right\rangle$$

Interpretable Tensor Contraction with Total-Variation Regularization

- Typically, we want to extract features from imaging data from **spatially-contiguous** regions.
- In tensor contraction, recall:

$$\mathcal{Z}^{(c)}(s,t) = \mathbf{A}(s,:)\mathcal{X}^{(c)}\left[\mathbf{B}(t,:)\right]^\top = \left\langle \mathbf{W}_{st}, \mathcal{X}^{(c)} \right\rangle$$

- To make $\mathbf{W}_{st}$ *sparse and smooth*, we introduce the anisotropic total-variation (TV) penalty [4] over $\mathbf{W}_{st}$:

$$\|\mathbf{W}_{st}\|_{\mathrm{TV}} = \|\nabla_x \mathbf{W}_{st}\|_1 + \|\nabla_y \mathbf{W}_{st}\|_1 \tag{4}$$

where ∇_x, ∇_y are gradient operators along the row and column direction.

Interpretable Tensor Contraction with Total-Variation Regularization

- Typically, we want to extract features from imaging data from **spatially-contiguous** regions.
- In tensor contraction, recall:

$$\mathcal{Z}^{(c)}(s,t) = \mathbf{A}(s,:)\mathcal{X}^{(c)}\left[\mathbf{B}(t,:)\right]^\top = \left\langle \mathbf{W}_{st}, \mathcal{X}^{(c)} \right\rangle$$

- To make $\mathbf{W}_{st}$ *sparse and smooth*, we introduce the anisotropic total-variation (TV) penalty [4] over $\mathbf{W}_{st}$:

$$\|\mathbf{W}_{st}\|_{\mathrm{TV}} = \|\nabla_x \mathbf{W}_{st}\|_1 + \|\nabla_y \mathbf{W}_{st}\|_1 \tag{4}$$

  where ∇_x, ∇_y are gradient operators along the row and column direction.
- Fortunately, the TV penalty has an elegant form under our tensor setup:

$$\sum_{s,t} \|\mathbf{W}_{st}\|_{\mathrm{TV}} = \|\mathbf{A}\|_1 \cdot \|\nabla_x \mathbf{B}\|_1 + \|\nabla_x \mathbf{A}\|_1 \cdot \|\mathbf{B}\|_1 \tag{5}$$

Complete Framework-Model

- Coupling tensor contraction with tensor GP, we end up with our **Tensor-GP** with **S**patial **T**ransformation (**Tensor-GPST**) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

$$f(.) = g \circ h(.) \sim \mathbf{GP}\left(0, \widetilde{K}(.,.)\right)$$

$$\widetilde{K}(\mathcal{X}_1, \mathcal{X}_2) = \mathrm{vec}\left(\mathcal{X}_1\right)^\top \left[\widetilde{\mathbf{U}}^\top \widetilde{\mathbf{U}}\right] \mathrm{vec}\left(\mathcal{X}_2\right)$$

where $\widetilde{\mathbf{U}} = \left(\mathbf{U}_3 \otimes \mathbf{U}_2 \otimes \mathbf{U}_1\right)\left(\mathbf{I}_C \otimes \mathbf{B} \otimes \mathbf{A}\right)$, g is the latent tensor GP, h is the tensor contraction.

Complete Framework-Estimation

- We estimate the kernel hyperparameters via penalized maximum marginal likelihood (Empirical Bayes):

$$\min_{(\boldsymbol{\theta},\boldsymbol{\eta},\sigma)} L\left(\boldsymbol{\theta},\boldsymbol{\eta},\sigma\right) = -\ell\left(\mathbf{y}|\boldsymbol{\theta},\boldsymbol{\eta},\sigma\right) + \lambda \sum_{s,t} \|\mathbf{W}_{st}\|_{\mathrm{TV}} \tag{6}$$

where $\boldsymbol{\theta} = \{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$, $\boldsymbol{\eta} = \{\mathbf{A}, \mathbf{B}\}$.

Complete Framework-Estimation

- We estimate the kernel hyperparameters via penalized maximum marginal likelihood (Empirical Bayes):

$$\min_{(\boldsymbol{\theta},\boldsymbol{\eta},\sigma)} L\left(\boldsymbol{\theta},\boldsymbol{\eta},\sigma\right) = -\ell\left(\mathbf{y}|\boldsymbol{\theta},\boldsymbol{\eta},\sigma\right) + \lambda \sum_{s,t} \|\mathbf{W}_{st}\|_{\mathrm{TV}} \tag{6}$$

  where $\boldsymbol{\theta} = \{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$, $\boldsymbol{\eta} = \{\mathbf{A}, \mathbf{B}\}$.
- More specifically:

$$\begin{aligned} L\left(\boldsymbol{\theta},\boldsymbol{\eta},\sigma\right) = \frac{1}{2}\ln\left|\mathbf{K}_{\boldsymbol{\theta},\boldsymbol{\eta}} + \sigma^2\mathbf{I}_n\right| + \frac{1}{2}\mathbf{y}^\top\left(\mathbf{K}_{\boldsymbol{\theta},\boldsymbol{\eta}} + \sigma^2\mathbf{I}_n\right)^{-1}\mathbf{y} \\ + \lambda\left(\|\mathbf{A}\|_1 \cdot \|\nabla_x\mathbf{B}\|_1 + \|\nabla_x\mathbf{A}\|_1 \cdot \|\mathbf{B}\|_1\right) \end{aligned}$$

  where $\mathbf{K}_{\boldsymbol{\theta},\boldsymbol{\eta}}$ is the $n \times n$ empirical gram matrix.

Algorithm: Block-Coordinate Proximal Gradient Descent

- We cyclically apply gradient-based updates on $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$ and σ.

Algorithm: Block-Coordinate Proximal Gradient Descent

- We cyclically apply gradient-based updates on $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$ and σ.
- As for $\mathbf{A}$ (similarly for $\mathbf{B}$), we further break down to two steps:
  1. Propose a gradient update $\widetilde{\mathbf{A}}$ via gradient descent:

$$\widetilde{\mathbf{A}} \leftarrow \mathbf{A} - \alpha \cdot \underbrace{\nabla_{\mathbf{A}} \ell \left(\mathbf{y} | \boldsymbol{\theta}, \boldsymbol{\eta}, \sigma\right)}_{\text{tractable thanks to the Woodbury identity}}$$

  2. The proximal step becomes multiple parallel *fused-lasso* [5] problem. For the s^{th} row of $\mathbf{A}$:

$$\widehat{\mathbf{A}}\left(s, :\right) = \arg\min_{\mathbf{x}} \frac{1}{2\alpha} \left\| \mathbf{x} - \widetilde{\mathbf{A}}\left(s, :\right) \right\|^2 + \left(\lambda \|\nabla_x \widehat{\mathbf{B}}\|_1\right) \cdot \|\mathbf{x}\|_1 + \left(\lambda \|\widehat{\mathbf{B}}\|_1\right) \cdot \|\nabla_x \mathbf{x}\|_1 \tag{7}$$

note how the *sparsity* and *smoothness* of $\mathbf{A}$ is regularized by the *smoothness* and *sparsity* of $\mathbf{B}$.

Algorithm: Block-Coordinate Proximal Gradient Descent

- We cyclically apply gradient-based updates on $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$ and σ.
- As for $\mathbf{A}$ (similarly for $\mathbf{B}$), we further break down to two steps:
  1. Propose a gradient update $\widetilde{\mathbf{A}}$ via gradient descent:

$$\widetilde{\mathbf{A}} \leftarrow \mathbf{A} - \alpha \cdot \underbrace{\nabla_{\mathbf{A}} \ell \left(\mathbf{y} | \boldsymbol{\theta}, \boldsymbol{\eta}, \sigma \right)}_{\text{tractable thanks to the Woodbury identity}}$$

  2. The proximal step becomes multiple parallel *fused-lasso* [5] problem. For the s^{th} row of $\mathbf{A}$:

$$\widehat{\mathbf{A}}(s,:) = \arg\min_{\mathbf{x}} \frac{1}{2\alpha} \left\| \mathbf{x} - \widetilde{\mathbf{A}}(s,:) \right\|^2 + \left(\lambda \| \nabla_x \widehat{\mathbf{B}} \|_1 \right) \cdot \| \mathbf{x} \|_1 + \left(\lambda \| \widehat{\mathbf{B}} \|_1 \right) \cdot \| \nabla_x \mathbf{x} \|_1 \tag{7}$$

  note how the *sparsity* and *smoothness* of $\mathbf{A}$ is regularized by the *smoothness* and *sparsity* of $\mathbf{B}$.
- We update one parameter at a time following the order:
  $\mathbf{A} \rightarrow \mathbf{B} \rightarrow \mathbf{U}_1 \rightarrow \mathbf{U}_2 \rightarrow \mathbf{U}_3 \rightarrow \sigma \rightarrow \mathbf{A} \rightarrow \ldots$ until convergence.

Algorithm: Convergence Analysis

- Under some mild conditions, after $(K+1)$ iterations, we have the following bounds on the loss function $L(\boldsymbol{\theta}, \boldsymbol{\eta}, \sigma)$ from its global minimum $L(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*, \sigma^*)$:

$$
\begin{aligned}
&4(K+1)\left(L\left(\widehat{\boldsymbol{\theta}}^{(K+1)}, \widehat{\boldsymbol{\eta}}^{(K+1)}, \widehat{\sigma}^{(K+1)}\right) - L(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*, \sigma^*)\right) \\
&\leq c^{-1}\delta^{(0)} && \text{(initialization error)} \\
&+ \sum_{k=0}^{K} h_\lambda\left(\|\widehat{\boldsymbol{\eta}}^{(K+1)} - \boldsymbol{\eta}^*\|_1\right) && \text{(due to TV Penalty)} \\
&+ c^{-1}\sum_{k=0}^{K} \tau\left(\|\widehat{\boldsymbol{\theta}}^{(K+1)} - \boldsymbol{\theta}^*\|_2, \|\widehat{\boldsymbol{\eta}}^{(K+1)} - \boldsymbol{\eta}^*\|_2\right) && \text{(due to coordinate descent)}
\end{aligned}
$$

Algorithm: Convergence Analysis

- Under some mild conditions, after $(K+1)$ iterations, we have the following bounds on the loss function $L(\boldsymbol{\theta}, \boldsymbol{\eta}, \sigma)$ from its global minimum $L(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*, \sigma^*)$:

$$
\begin{aligned}
&4(K+1)\left(L\left(\widehat{\boldsymbol{\theta}}^{(K+1)}, \widehat{\boldsymbol{\eta}}^{(K+1)}, \widehat{\sigma}^{(K+1)}\right) - L\left(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*, \sigma^*\right)\right) \\
&\leq c^{-1}\delta^{(0)} && \text{(initialization error)} \\
&+ \sum_{k=0}^{K} h_\lambda\left(\|\widehat{\boldsymbol{\eta}}^{(K+1)} - \boldsymbol{\eta}^*\|_1\right) && \text{(due to TV Penalty)} \\
&+ c^{-1}\sum_{k=0}^{K} \tau\left(\|\widehat{\boldsymbol{\theta}}^{(K+1)} - \boldsymbol{\theta}^*\|_2, \|\widehat{\boldsymbol{\eta}}^{(K+1)} - \boldsymbol{\eta}^*\|_2\right) && \text{(due to coordinate descent)}
\end{aligned}
$$

- This result states that the algorithm converges to a local minimum at a rate of $\mathcal{O}(1/K)$, and we confirmed this empirically.

Application to Solar Flare Intensity Forecasting

- Model:

$$y_i = g \circ h\left(\mathcal{X}_i\right) + \epsilon \tag{8}$$

  - y_i: solar flare intensity
  - $\mathcal{X}_i$: $(H, W, C) = (50, 50, 10)$ AIA-HMI imaging dataset
  - $n = 1,329$ samples of M/X-class ($n_{\mathrm{M/X}} = 479$) and B-class ($n_{\mathrm{B}} = 850$).
  - set the contracted tensor size as $3 \times 3 \times 10$.
  - chronologically splits the data into train/test.

Application to Solar Flare Intensity Forecasting

- Model:

$$y_i = g \circ h\left(\mathcal{X}_i\right) + \epsilon \tag{8}$$

  - y_i: solar flare intensity
  - $\mathcal{X}_i$: $(H, W, C) = (50, 50, 10)$ AIA-HMI imaging dataset
  - $n = 1,329$ samples of M/X-class ($n_{\mathrm{M/X}} = 479$) and B-class ($n_{\mathrm{B}} = 850$).
  - set the contracted tensor size as $3 \times 3 \times 10$.
  - chronologically splits the data into train/test.
- g: the Tensor Gaussian Process on the $3 \times 3 \times 10$ latent tensors.
- h: the Tensor Contraction layer for dimensionality reduction.

Channel Average Tensor: B-class Solar Flare

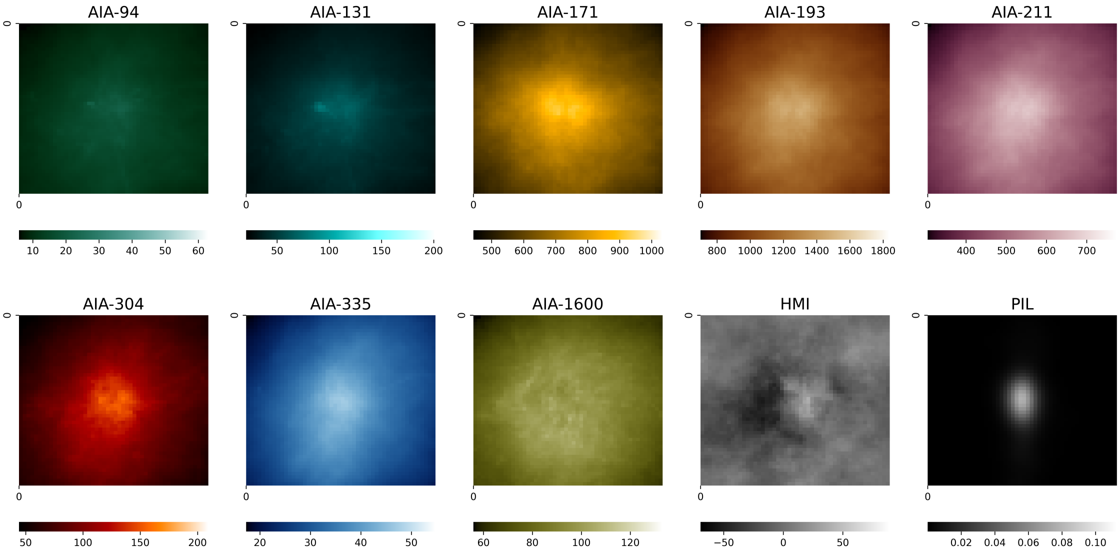


Figure: Channel-wise Average for all B-class flares, each image is of size 50×50.

Channel Average Tensor: M-class Solar Flare

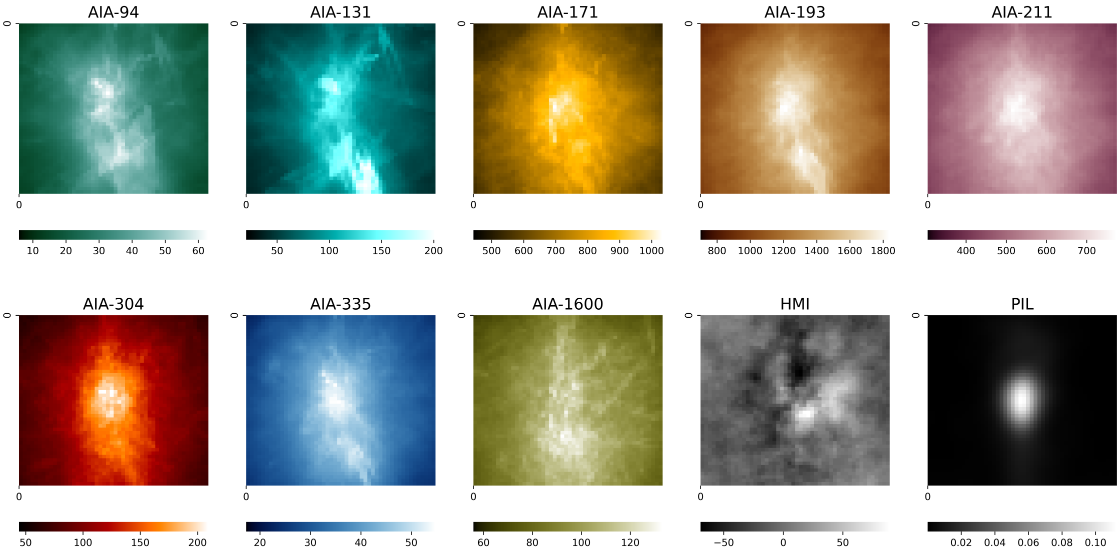


Figure: Channel-wise Average for all M/X-class flares, each image is of size 50×50.

Fitted Parameter for $\widehat{h}$ (Tensor Contraction)

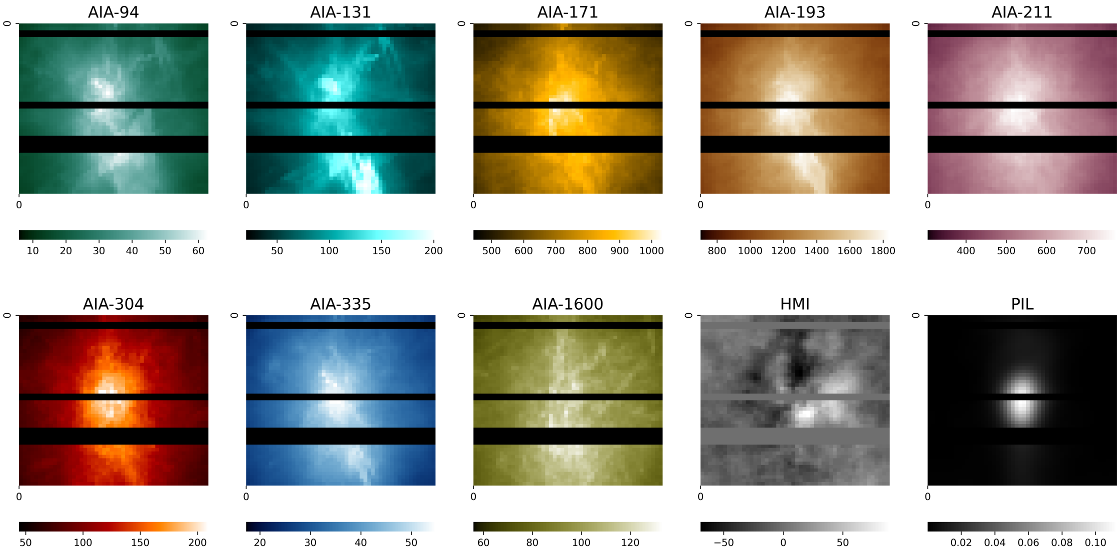


Figure: Pixels with non-zero tensor contraction weights. Plotted with M-class channel average.

Fitted Parameter for $\widehat{h}$ (Tensor Contraction)

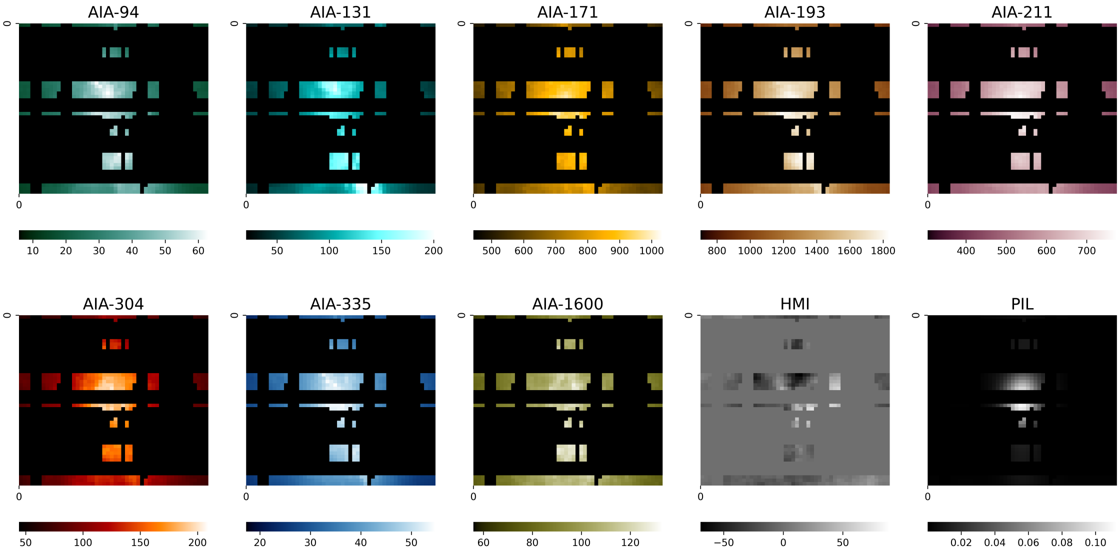
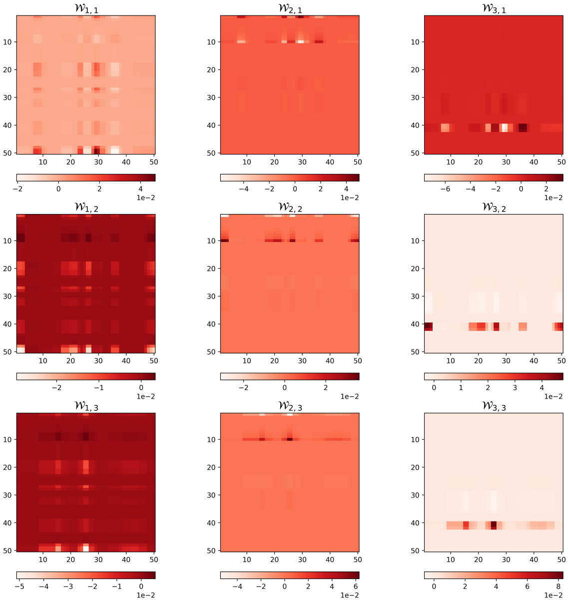


Figure: Pixels with tensor contraction weights $> 5 \times 10^{-3}$. Plotted with M-class channel average.

Fitted Parameter for $\widehat{h}$ (Tensor Contraction)



Variance Decomposition for $\widehat{g}$ (GP)

- Recall that the multi-linear kernel of $g(.) \sim \mathbf{GP}\left(0, K(.,.)\right)$ is:

$$K(\widehat{h}(\mathcal{X}_1), \widehat{h}(\mathcal{X}_2)) = \text{vec}\left(\widehat{h}(\mathcal{X}_1)\right)^\top [\mathbf{K}_3 \otimes \mathbf{K}_2 \otimes \mathbf{K}_1] \, \text{vec}\left(\widehat{h}(\mathcal{X}_2)\right) \tag{9}$$

Variance Decomposition for $\widehat{g}$ (GP)

- Recall that the multi-linear kernel of $g(.) \sim \mathbf{GP}\left(0, K(.,.)\right)$ is:

$$K(\widehat{h}(\mathcal{X}_1), \widehat{h}(\mathcal{X}_2)) = \operatorname{vec}\left(\widehat{h}(\mathcal{X}_1)\right)^\top [\mathbf{K}_3 \otimes \mathbf{K}_2 \otimes \mathbf{K}_1] \operatorname{vec}\left(\widehat{h}(\mathcal{X}_2)\right) \tag{9}$$

- Equivalently:

$$\operatorname{Cov}(y_1, y_2) =$$

$$\sum_{\substack{(s_1,t_1,c_1)\\(s_2,t_2,c_2)}}^{h,w,C} \underbrace{\mathbf{K}_1(s_1, s_2) \cdot \mathbf{K}_2(t_1, t_2)}_{\text{Feature Map Importance}} \times \overbrace{\mathbf{K}_3(c_1, c_2)}^{\text{Channel Importance}} \times \underbrace{\left\langle \mathbf{W}_{s_1,t_1}, \mathcal{X}_1^{(c_1)} \right\rangle \cdot \left\langle \mathbf{W}_{s_2,t_2}, \mathcal{X}_2^{(c_2)} \right\rangle}_{\text{Latent Features Similarity}}$$

$$+ \delta_{12} \cdot \underbrace{\sigma^2}_{\text{Noise}}$$

Fitted Parameter for $\widehat{g}$ (GP)

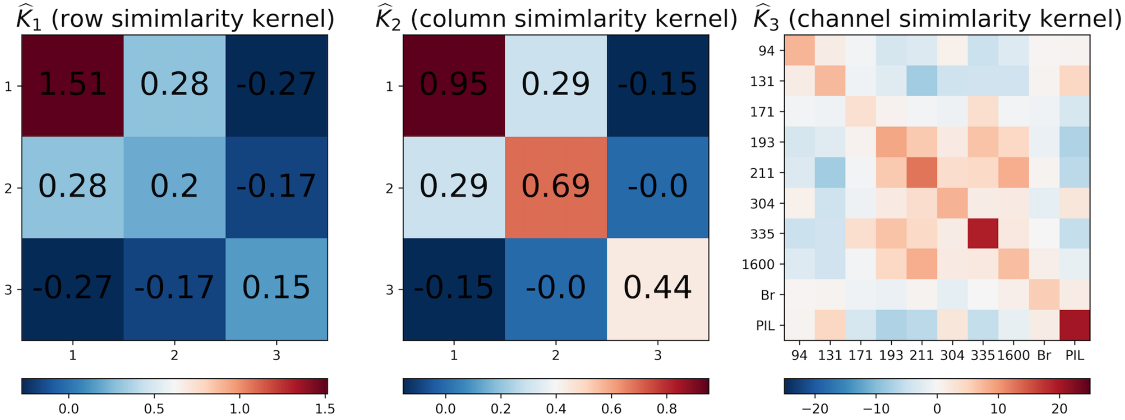


Figure: Estimates for $\mathbf{K}_1, \mathbf{K}_2, \mathbf{K}_3$ of the latent tensor GP $\widehat{g}$.

Fitted Parameter for $\widehat{g}$ (GP)

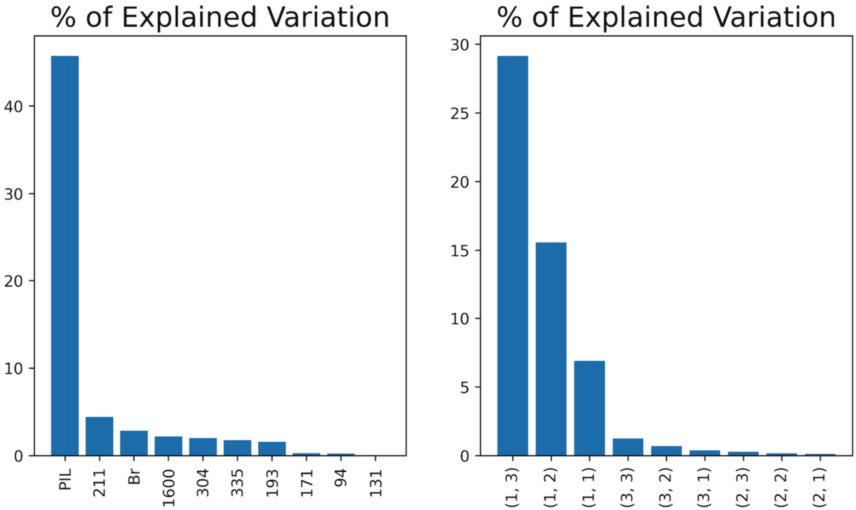


Figure: Channel (left) and feature map (right) % of explained variation of the latent tensor GP $\widehat{g}$.

Tensor Regression Result: Chronological Split

Model	MSE	R^2	P_{cover}	TSS	$\widehat{\sigma}$
Tensor-GP	0.405 0.978	0.374 0.184	0.969 0.920	0.511 0.309	0.662
Tensor-GPST ($\lambda = 0.1$)	**0.392** 0.772	**0.394** 0.220	0.968 0.900	**0.518** 0.366	0.634
Tensor-GPST ($\lambda = 0.5$)	0.429 **0.611**	0.337 0.269	**0.970** **0.957**	0.448 **0.432**	0.661
Tensor-GPST ($\lambda = 1$)	0.414 0.720	0.361 0.235	0.960 0.925	0.476 0.338	0.649
CP	0.452 0.648	0.303 **0.310**	–	0.438 0.400	–
Tucker	0.462 0.655	0.287 0.301	–	0.428 0.400	–

Table: Training (top) and testing (bottom) performances. Metrics: MSE: Mean-Squared Error; R^2: R-squared; P_{cover}: coverage probability; TSS: True Skill Statistics.

Future Research Topics

- Identifiability issue between $g(.)$ and $h(.)$.
- Scalable GP regression with stochastic variational inference.
- Enable the model to handle binary responses.
- Account for image transformation (e.g. rotate, shift, shear) invariance in the tensor kernel.

Summary of the Talk

In this talk, we:

- propose a scalar-on-tensor Gaussian Process Regression (GPR) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

where:

Summary of the Talk

In this talk, we:

- propose a scalar-on-tensor Gaussian Process Regression (GPR) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

  where:

  - $h(.)$: condense the tensor data to a latent tensor via tensor contraction.

Summary of the Talk

In this talk, we:

- propose a scalar-on-tensor Gaussian Process Regression (GPR) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

  where:

  - $h(.)$: condense the tensor data to a latent tensor via tensor contraction.
  - $g(.)$: use multi-linear kernel to do GPR in the latent tensor space.

Summary of the Talk

In this talk, we:

- propose a scalar-on-tensor Gaussian Process Regression (GPR) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

  where:
  - $h(.)$: condense the tensor data to a latent tensor via tensor contraction.
  - $g(.)$: use multi-linear kernel to do GPR in the latent tensor space.
- introduce an ℓ_1 Total-Variation (TV) Penalty over $h(.)$ for interpretable tensor dimension reduction, and propose a coordinate proximal gradient descent method for estimation.

Summary of the Talk

In this talk, we:

- propose a scalar-on-tensor Gaussian Process Regression (GPR) model:

$$y = g \circ h(\mathcal{X}) + \epsilon$$

  where:
  - $h(.)$: condense the tensor data to a latent tensor via tensor contraction.
  - $g(.)$: use multi-linear kernel to do GPR in the latent tensor space.
- introduce an ℓ_1 Total-Variation (TV) Penalty over $h(.)$ for interpretable tensor dimension reduction, and propose a coordinate proximal gradient descent method for estimation.
- demonstrate the effectiveness via a solar flare intensity forecasting application.

References I

[1] H. Zhou, L. Li, and H. Zhu, "Tensor Regression with Applications in Neuroimaging Data Analysis," *Journal of the American Statistical Association*, vol. 108, no. 502, pp. 540–552, 2013.

[2] X. Li, D. Xu, H. Zhou, and L. Li, "Tucker Tensor Regression and Neuroimaging Analysis," *Statistics in Biosciences*, vol. 10, no. 3, pp. 520–545, 2018.

[3] R. Yu, G. Li, and Y. Liu, "Tensor Regression meets Gaussian Processes," in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2018, pp. 482–490.

[4] X. Wang, H. Zhu, and A. D. N. Initiative, "Generalized Scalar-on-Image Regression Models via Total Variation," *Journal of the American Statistical Association*, vol. 112, no. 519, pp. 1156–1168, 2017.

[5] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani, "Pathwise Coordinate Optimization," *The Annals of Applied Statistics*, vol. 1, no. 2, pp. 302–332, 2007.