Video Imputation and Prediction Models in Context of Space Weather Monitoring[1]

Hu Sun [1]

Zhijun Hua[1], Jiaen Ren[2], Shasha Zou[2], Yuekai Sun[1], Yang Chen[1]

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

[2]Department of Climate and Space Sciences and Engineering, University of Michigan, Ann Arbor

April 27, 2021

[1]Click here for the preprint version of our paper.

Overview

1. Background
    - Total Electron Content (TEC) map
    - Matrix Completion Problem
    - Spherical Harmonics
2. Method
    - Conceptual Framework
    - Algorithm
    - Theoretical Guarantees
3. Empirical Analysis
    - Simulation Study
    - Imputing TEC map
4. Conclusion & Future Plan

Total Electron Content (TEC) map

- Ionosphere Total Electron Content (TEC) is defined as the total number of electrons in the path between satellite[2] radio transmitter and ground-based receiver. (1 TEC unit (TECU) $= 10^{16}$ electrons/m²)

[2] satellite of The Global Navigation Satellite Systems (GNSS)

Total Electron Content (TEC) map

- Ionosphere Total Electron Content (TEC) is defined as the total number of electrons in the path between satellite[2] radio transmitter and ground-based receiver. (1 TEC unit (TECU) = 10^{16} electrons/m²)
- TEC affects the propagation of radio waves, leading up to 10s meters positioning error in the GNSS Positioning, Navigation and Timing (PNT) services. Better knowledge of TEC map will make PNT services more accurate.

[2] satellite of The Global Navigation Satellite Systems (GNSS)

Total Electron Content (TEC) map

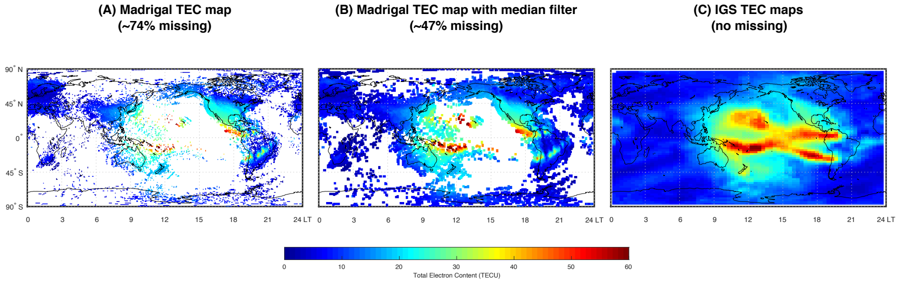


Figure: TEC map from the Madrigal Database (A) without median filter, (B) with a 3° × 3° median filter and (C) TEC map from the International GNSS Service (IGS).

Total Electron Content (TEC) map

- The goal of the project is to reasonably "fill in" the missing values within TEC maps. Pan et al. (2020) used DCGAN-based models for TEC map completion, relying on IGS TEC maps as either reference or training data. But overall the IGS data is of low-resolution, and we want to preserve the **high-resolution nature** of the TEC map.

Matrix Completion Problem

- To impute the TEC maps, we adopt classical statistics techniques called **matrix completion**.

Matrix Completion Problem

- To impute the TEC maps, we adopt classical statistics techniques called **matrix completion**.
- Matrix completion is a commonly used method in designing recommender systems. With a user-item rating matrix, for example, matrix completion can infer the potential rating a user would give to an item he/she has never consumed.

Matrix Completion Problem

Rank-Restricted SVD (Mazumder et al., 2010)

$$\min_{M_t} \quad H(M_t) := \frac{1}{2}\|\mathrm{P}_{\Omega_t}(X_t - M_t)\|_F^2 + \lambda\|M_t\|_* \tag{1}$$

where $\|M_t\|_*$ is the nuclear norm, i.e. sum of all singular values, of M_t. Following the notation in (Candès and Tao, 2010), the projection $\mathrm{P}_{\Omega_t}(X_t)$ is an $m \times n$ matrix keeping all observed entries of X_t and replacing all missing entries with 0.

- It is a well-known result that the solution is $M_t = U_r \boldsymbol{S}_\lambda(D_r) V_r^T$, where $r = \min(m, n)$ and U_r, D_r, V_r are the components of rank-r SVD of X_t. $\boldsymbol{S}_\lambda(D_r) = \text{diag}\left[(\sigma_1 - \lambda)_+, (\sigma_2 - \lambda)_+, \ldots, (\sigma_r - \lambda)_+\right]$ is the soft-thresholding operator.

Matrix Completion with Factorization

Maximum-margin Matrix Factorization (MMMF) (Srebro et al., 2005)

$$\min_{A_t, B_t} \quad F(A_t, B_t) := \frac{1}{2}\|\mathrm{P}_{\Omega_t}(X_t - A_t B_t^T)\|_F^2 + \frac{\lambda_1}{2}(\|A_t\|_F^2 + \|B_t\|_F^2) \quad (2)$$

with solution $\widehat{A}_t = U_r \boldsymbol{S}_\lambda(D_r)^{\frac{1}{2}}$ and $\widehat{B}_t = V_r \boldsymbol{S}_\lambda(D_r)^{\frac{1}{2}}$

Matrix Completion with Factorization

Maximum-margin Matrix Factorization (MMMF) (Srebro et al., 2005)

$$\min_{A_t, B_t} \quad F(A_t, B_t) := \frac{1}{2}\|\mathrm{P}_{\Omega_t}(X_t - A_t B_t^T)\|_F^2 + \frac{\lambda_1}{2}(\|A_t\|_F^2 + \|B_t\|_F^2) \quad (2)$$

with solution $\widehat{A}_t = U_r \boldsymbol{S}_\lambda(D_r)^{\frac{1}{2}}$ and $\widehat{B}_t = V_r \boldsymbol{S}_\lambda(D_r)^{\frac{1}{2}}$

- Such a factorization setup has direct interpretations in the factor matrices A, B. For example, the original map is of size $m \times n$, where m, n corresponds to latitude and longitude. Then each row in A and B can be considered as the "latent feature" of each latitude and longitude. The final imputation at any location is the inner product of the feature vectors of the corresponding latitude and longitude.

Matrix Completion with Factorization

SoftImpute-Alternating Least Square (Hastie et al., 2015)

$$\min_{A_t, B_t} \quad F(A_t, B_t) := \frac{1}{2}\|\widehat{X}_t - A_t B_t^T\|_F^2 + \frac{\lambda_1}{2}(\|A_t\|_F^2 + \|B_t\|_F^2) \tag{2}$$

where $\widehat{X}_t$ is a "filled-in" $m \times n$ matrix, with $\widehat{X}_t = \mathrm{P}_{\Omega_t}(X_t) + \mathrm{P}_{\Omega_t^\perp}(\tilde{A}_t \tilde{B}_t^T)$, and $\tilde{A}_t, \tilde{B}_t$ are the two factor matrices in the previous iterative step.

Matrix Completion with Factorization

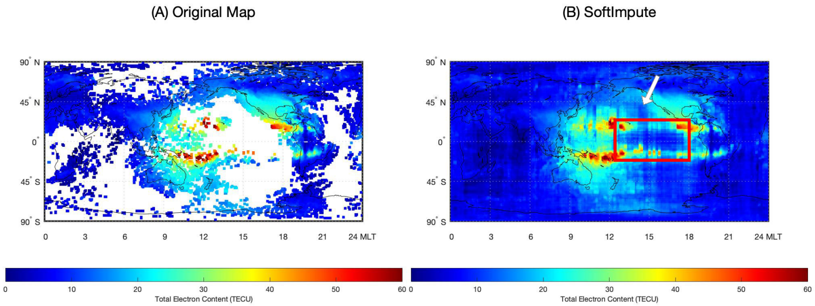


Figure: TEC maps: observed (left) and fitted by the SoftImpute approach (right).

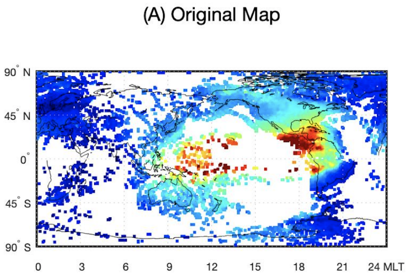
Spherical Harmonics

- Apart from the matrix completion method, one can also impute each TEC map X_t with **Spherical Harmonics (SH)**.

Spherical Harmonics

- Apart from the matrix completion method, one can also impute each TEC map X_t with **Spherical Harmonics (SH)**.
- Spherical Harmonics is approximating data on a surface with a linear combination of several basis functions. For TEC map, we can think of TEC value distributed on the globe, and we use Spherical Harmonics to approximate this surface of TEC values.

Spherical Harmonics



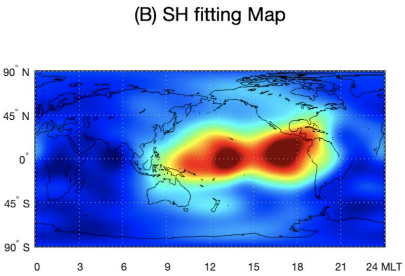


Figure: Example of Spherical Harmonics Fitting

Spherical Harmonics

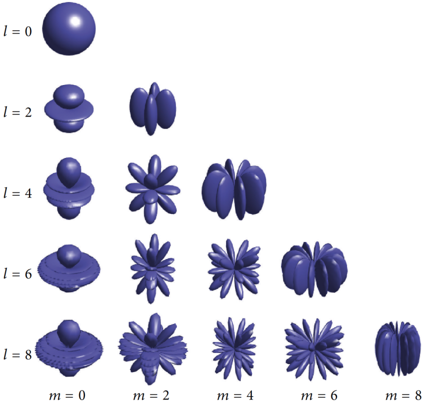


FIGURE 1: Spherical harmonics.

Figure: Source: Nortje et al., 2015

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)
- Use a matrix factorization formulation as the imputed matrix

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)
- Use a matrix factorization formulation as the imputed matrix
- Use spherical harmonics as a warm-start (we call it “auxiliary data“)

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)
- Use a matrix factorization formulation as the imputed matrix
- Use spherical harmonics as a warm-start (we call it “auxiliary data“)
- Penalizes the matrix norm of the factor matrices (soft constraint on rank)

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)
- Use a matrix factorization formulation as the imputed matrix
- Use spherical harmonics as a warm-start (we call it “auxiliary data“)
- Penalizes the matrix norm of the factor matrices (soft constraint on rank)
- Reinforce smoothness of the imputed results along the temporal dimension

Conceptual Framework

Our final framework has the following features:

- Impute a consecutive sequence of TEC maps (i.e. TEC videos)
- Use a matrix factorization formulation as the imputed matrix
- Use spherical harmonics as a warm-start (we call it "auxiliary data")
- Penalizes the matrix norm of the factor matrices (soft constraint on rank)
- Reinforce smoothness of the imputed results along the temporal dimension
- Objective function has the form:

$$
\begin{aligned}
\text{Imputation Loss} &+ \lambda_1 \times \text{Matrix Norm Penalty} \\
&+ \lambda_2 \times \text{Temporal Smoothness Penalty} \\
&+ \lambda_3 \times \text{Auxiliary Data Penalty}
\end{aligned}
$$

Conceptual Framework

Our model has a name "Video Imputation with SoftImpute, Temporal smoothing and Auxiliary data" (**VISTA**)

Conceptual Framework

Objective Function

$$
\begin{aligned}
\min_{A_{1:T}, B_{1:T}} \Bigg\{ F(A_{1:T}, B_{1:T}) \triangleq\ & \frac{1}{2} \sum_{t=1}^{T} \| P_{\Omega_t}(X_t - A_t B_t^T) \|_F^2 \\
& + \frac{\lambda_1}{2} \sum_{t=1}^{T} (\|A_t\|_F^2 + \|B_t\|_F^2) \\
& + \frac{\lambda_2}{2} \sum_{t=2}^{T} \|A_t B_t^T - A_{t-1} B_{t-1}^T\|_F^2 \\
& + \frac{\lambda_3}{2} \sum_{t=1}^{T} \|Y_t - A_t B_t^T\|_F^2 \Bigg\}
\end{aligned}
$$

where $Y_1, Y_2, \ldots, Y_T$ are $m \times n$ auxiliary data with no missing values.

Conceptual Framework

An alternative perspective to interpret the objective function is to think about it under a Bayesian setup:

$$
\begin{aligned}
X_t &\sim N(A_t B_t^T, \sigma^2) && \text{(Data generating model)} \\
A_t &\sim N(0, \frac{1}{\lambda_1}\sigma^2) && \text{(Prior of A)} \\
B_t &\sim N(0, \frac{1}{\lambda_1}\sigma^2) && \text{(Prior of B)} \\
A_t B_t^T &\sim N(A_{t-1} B_{t-1}^T, \frac{1}{\lambda_2}\sigma^2) && \text{(Random walk assumption)} \\
A_t B_t^T &\sim N(Y_t, \frac{1}{\lambda_3}\sigma^2) && \text{(Prior of } AB^T\text{)}
\end{aligned}
$$

And the objective function is maximizing the posterior likelihood based on T frames of data.

Algorithm Outline

- There are in total T frames to be imputed at the same time, and each frame has its own A_t, B_t factors.

Algorithm Outline

- There are in total T frames to be imputed at the same time, and each frame has its own A_t, B_t factors.
- Update the factors $A_1, A_2, \ldots, A_T, B_1, B_2, \ldots, B_T$ cyclically: $A_1 \to A_2 \to \cdots \to A_T \to B_1 \to B_2 \to \cdots \to B_T \to A_1 \to A_2 \to \ldots$.

Algorithm Outline

- There are in total T frames to be imputed at the same time, and each frame has its own A_t, B_t factors.
- Update the factors $A_1, A_2, \ldots, A_T, B_1, B_2, \ldots, B_T$ cyclically: $A_1 \to A_2 \to \cdots \to A_T \to B_1 \to B_2 \to \cdots \to B_T \to A_1 \to A_2 \to \ldots$.
- Fix $2\mathrm{T} - 1$ matrices and update one matrix at a time with majorization-minimization (MM) algorithm. The final form is simply doing a least square.

Update Matrix with Least Square

Suppose in the k-th round, we wish to update A_t. The current values for the other factors are: $A_1^{(k+1)}, A_2^{(k+1)}, \dots, A_{t-1}^{(k+1)}, A_t^{(k)}, \dots A_T^{(k)}$ and $B_1^{(k)}, B_2^{(k)}, \dots, B_T^{(k)}$. Keeping every matrix other than A_t fixed at their current values, the convex optimization problem is reduced to the following optimization problem:

$$
\begin{aligned}
&\min_{A_t} \Big\{ Q(A_t | A_{1:t-1}^{(k+1)}, A_{t+1:T}^{(k)}, B_{1:T}^{(k)}) \\
&\triangleq \frac{1}{2} \| P_{\Omega_t}(X_t - A_t (B_t^{(k)})^T) \|_F^2 + \frac{\lambda_1}{2} \|A_t\|_F^2 + \frac{\lambda_3}{2} \| Y_t - A_t (B_t^{(k)})^T \|_F^2 \\
&\quad + \frac{\lambda_2}{2} \mathrm{I}_{\{t>1\}} \| A_t (B_t^{(k)})^T - A_{t-1}^{(k+1)} (B_{t-1}^{(k)})^T \|_F^2 \\
&\quad + \frac{\lambda_2}{2} \mathrm{I}_{\{t<T\}} \| A_{t+1}^{(k)} (B_{t+1}^{(k)})^T - A_t (B_t^{(k)})^T \|_F^2 \Big\}
\end{aligned}
$$

Update Matrix with Least Square

The very first term is $\|P_{\Omega_t}(X_t - A_t(B_t^{(k)})^T)\|_F^2$, which can be upper bounded easily by:

$$\|P_{\Omega_t}(X_t - A_t(B_t^{(k)})^T)\|_F^2 \leq \|P_{\Omega_t}(X_t) + P_{\Omega_t^{\perp}}(A_t^{(k)}(B_t^{(k)})^T) - A_t(B_t^{(k)})^T\|_F^2$$

Update Matrix with Least Square

Substituting the first term with its upper bound, and denote the new objective function as $\tilde{Q}(A_t|A_{1:t-1}^{(k+1)}, A_{t+1:T}^{(k)}, B_{1:T}^{(k)})$. Then one can take the derivative of $\tilde{Q}$ w.r.t. A_t and sets it to zero and get:

$$A_t^{(k+1)} = \left[(1 + \lambda_2(\mathsf{I}_{\{t<T\}} + \mathsf{I}_{\{t>1\}}) + \lambda_3)(B_t^{(k)})^T B_t^{(k)} + \lambda_1 \mathrm{I}\right]^{-1} Z_t^{(k)} B_t^{(k)}$$

where

$$\begin{aligned} Z_t^{(k)} = {} & P_{\Omega_t}(X_t) + P_{\Omega_t^{\perp}}(A_t^{(k)}(B_t^{(k)})^T \\ & + \lambda_2 \left(\mathsf{I}_{\{t>1\}} A_{t-1}^{(k+1)}(B_{t-1}^{(k)})^T + \mathsf{I}_{\{t<T\}} A_{t+1}^{(k)}(B_{t+1}^{(k)})^T\right) \\ & + \lambda_3 Y_t \end{aligned}$$

Final Algorithm

Algorithm 1 softImpute-ALS with Temporal Smoothing and Auxiliary Data

Input: $m \times n$ Sparse data $X_1, X_2, \dots, X_T$, $m \times n$ auxiliary data $Y_1, Y_2, \dots, Y_T$, operating rank r. Maximum iteration K and convergence threshold τ.

Output: Imputation of sparse data $A_1 B_1^T, A_2 B_2^T, \dots, A_T B_T^T$.

Initialization: For $1 \le t \le T$, $A_t^{(1)} = U_t D_t, B_t^{(1)} = V_t D_t$, where U_t, V_t are $m \times r, n \times r$ randomly chosen matrix with orthogonal columns. D_t is $\mathrm{I}_{r \times r}$

Update A:

for $t = 1 : T$ **do**

  a. Let $X_t^{(k)} = P_{\Omega_t}(X_t) + P_{\Omega_t^{\perp}}(A_t^{(k)}(B_t^{(k)})^T)$, which is the "filled-in" version of X_t

  b. Let $Z_t^{(k)}$ be the weighted label in equation (11)

  c. $A_t^{(k+1)}$ is updated as equation (13)

end for

Update B: For every t, repeat a,b,c steps above, with $X_t^{(k)}, Z_t^{(k)}$ being replace by $X_t^{(k+\frac{1}{2})}, Z_t^{(k+\frac{1}{2})}$. $B_t^{(k+1)}$ is calculated following equation (14)

Repeat updating $A_{1:T}$ and $B_{1:T}$ until convergence. The algorithm converges when $\max\{\nabla F_1^{(k)}, \nabla F_2^{(k)}, \dots, \nabla F_T^{(k)}\} < \tau$, with $\nabla F_t^{(k)}$ defined in (15).

For any t, denote the final output as A_t^*, B_t^*. Let $X_t^* = P_{\Omega_t}(X_t) + P_{\Omega_t^{\perp}}(A_t^*(B_t^*)^T)$.

  Do SVD for $A_t^*(B_t^*)^T = U_t^*(D_t^*)^2(V_t^*)^T$

  Define $M_t = X_t^* V_t^*$ and do SVD for $M_t = \tilde{U}_t \tilde{D}_t R_t^T$.

  Do soft-thresholding on $\tilde{D}_t$: $\tilde{D}_{t,\lambda_1} = \textbf{diag}\left[(\sigma_1 - \lambda_1)_+, (\sigma_2 - \lambda_1)_+, \dots, (\sigma_r - \lambda_1)_+\right]$

  Output imputation for time t as $\tilde{U}_t \tilde{D}_{t,\lambda_1}(V_t^* R_t)^T$

Convergence Guarantee

Across the iterations of our algorithm, we denote the iterative value of $A_{1:T}, B_{1:T}$ in the k-th round of algorithm as $A_{1:T}^{(k)}, B_{1:T}^{(k)}$. Then we can prove the following property of our algorithm:

Convergence Guarantee

Across the iterations of our algorithm, we denote the iterative value of $A_{1:T}, B_{1:T}$ in the k-th round of algorithm as $A_{1:T}^{(k)}, B_{1:T}^{(k)}$. Then we can prove the following property of our algorithm:

Objective Function is Non-Increasing

Define the descent of objective function value at iteration k as $\Delta_k = F(A_{1:T}^{(k)}, B_{1:T}^{(k)}) - F(A_{1:T}^{(k+1)}, B_{1:T}^{(k+1)})$. Then the value of the objective function is non-increasing, i.e.,

$$F(A_{1:T}^{(k)}, B_{1:T}^{(k)}) \geq F(A_{1:T}^{(k+1)}, B_{1:T}^{(k)}) \geq F(A_{1:T}^{(k+1)}, B_{1:T}^{(k+1)}),$$

thus $\Delta_k \geq 0$, for all $k \geq 1$.

Convergence Rate

Convergence Rate Lower Bound

Let the limit of the objective function $F(A_{1:T}^{(k)}, B_{1:T}^{(k)})$ be f^{∞}, we have:

$$\min_{1\le k\le K} \Delta_k \le \frac{F(A_{1:T}^{(1)}, B_{1:T}^{(1)}) - f^{\infty}}{K}$$

where K is the total number of iterations.

These results suggest that our algorithm is converging at a rate of $O(1/K)$.

Empirical Analysis: Data Pipeline

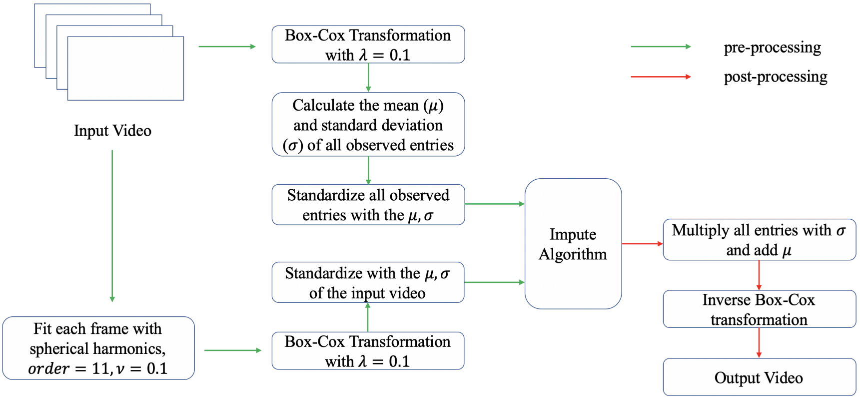


Figure: Data Pipeline

Simulation Study: Data

- In our simulation study, we use the IGS dataset of TEC map, which is of low resolution but is fully observed without missing values. We fit our model on several days of IGS data in Sept. 2017. Each day contains data of size $181 \times 361 \times 96$, where every matrix is of size 181×361.

Simulation Study: Data

- In our simulation study, we use the IGS dataset of TEC map, which is of low resolution but is fully observed without missing values. We fit our model on several days of IGS data in Sept. 2017. Each day contains data of size $181 \times 361 \times 96$, where every matrix is of size 181×361.
- To mimic some data missing patterns typically observed in Madrigal database (high-res TEC maps), we artificially "created" some missingness.

Simulation Study: Missingness Design

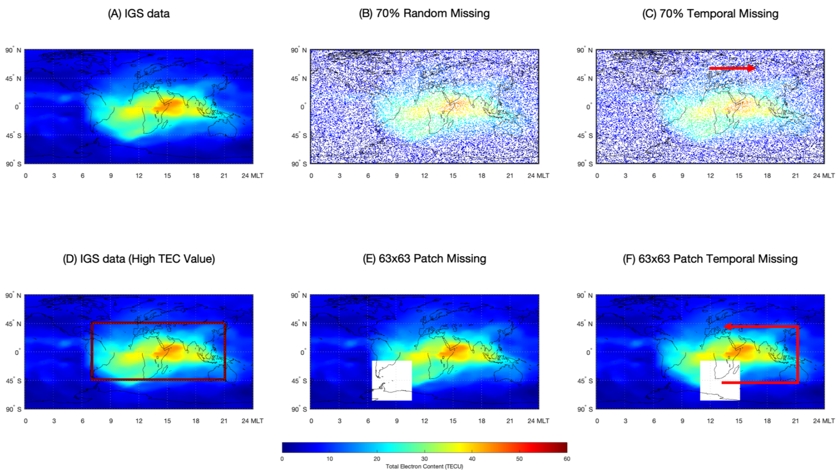


Figure: Create Missing Data

Simulation Study: Missingness Design

- Random missingness (sub-figure B): for each matrix, randomly drop 30%/50%/70% of the pixels.

Simulation Study: Missingness Design

- Random missingness (sub-figure B): for each matrix, randomly drop 30%/50%/70% of the pixels.
- Temporal missingness (sub-figure C): for the first matrix, randomly drop 30%/50%/70% of pixels, and let the missing mask move 6 columns horizontally (direction shown as the red arrow).

Simulation Study: Missingness Design

- Random missingness (sub-figure B): for each matrix, randomly drop 30%/50%/70% of the pixels.
- Temporal missingness (sub-figure C): for the first matrix, randomly drop 30%/50%/70% of pixels, and let the missing mask move 6 columns horizontally (direction shown as the red arrow).
- Random patch missingness (sub-figure E): for each frame, randomly pick a center on a fixed bounding box around high TEC value region (sub-figure D) and create a 27×27 or 45×45 or 63×63 patch as missing.

Simulation Study: Missingness Design

- Random missingness (sub-figure B): for each matrix, randomly drop 30%/50%/70% of the pixels.
- Temporal missingness (sub-figure C): for the first matrix, randomly drop 30%/50%/70% of pixels, and let the missing mask move 6 columns horizontally (direction shown as the red arrow).
- Random patch missingness (sub-figure E): for each frame, randomly pick a center on a fixed bounding box around high TEC value region (sub-figure D) and create a 27×27 or 45×45 or 63×63 patch as missing.
- Temporal patch missingness (sub-figure F): similar to patch missingness, but the center of the $27 \times 27/45 \times 45/63 \times 63$ patch moves along the bounding box at the speed of 6 columns(rows) per matrix (anti-clockwise as shown by the red arrow).

Simulation Study: Models & Metrics

We consider fitting the following VISTA models on each of the missing pattern:

1. **soft**: softImpute as in Hastie et al., 2015: $\lambda_1 = 0.9, \lambda_2 = 0, \lambda_3 = 0$. (**Benchmark model**)
2. **TS**: softImpute + temporal smoothing: $\lambda_1 = 0.9, \lambda_2 = 0.05, \lambda_3 = 0$.
3. **SH**: softImpute + auxiliary data based on spherical harmonics: $\lambda_1 = 0.9, \lambda_2 = 0, \lambda_3 = 0.01$.
4. **TS+SH**: softImpute + temporal smoothing + auxiliary data based on spherical harmonics: $\lambda_1 = 0.9, \lambda_2 = 0.05, \lambda_3 = 0.01$.

Simulation Study: Models & Metrics

To evaluate the performance of the imputation, we compute **Relative Squared Error** (RSE):

$$\mathrm{RSE}(X_t, X_t^*, \Omega_t) = \frac{\|\mathrm{P}_{\Omega_t^\perp}(X_t^* - X_t)\|_F}{\|\mathrm{P}_{\Omega_t^\perp}(X_t)\|_F},$$

where X_t is the fully-observed IGS data. Ω_t is the bitmap indicating the observed pixels. $\mathrm{P}_{\Omega_t^\perp}(.)$ is a projection operator onto the missing pixels. X_t^* is the imputation of $\mathrm{P}_{\Omega_t}(X_t)$ and $\|.\|_F$ is the Frobenius norm.

Simulation Study: Result of Random Missingness

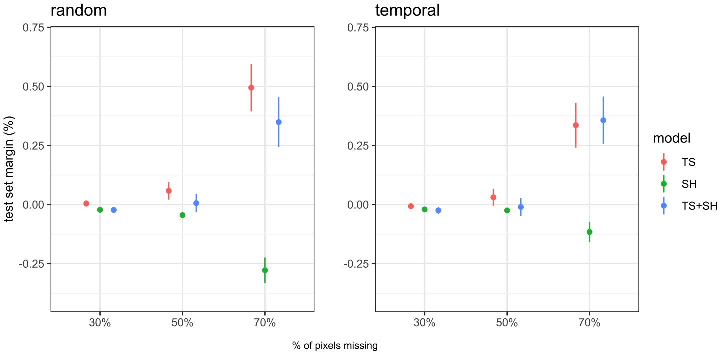


Figure: Random missing and temporal missing results. Three variants of our method are considered: TS, SH, TS+SH. The scatter points show the average test set RSE margin over baseline softImpute method, positive means performance better than softImpute. Error bar gives the 95% confidence interval.

Simulation Study: Result of Patch Missingness

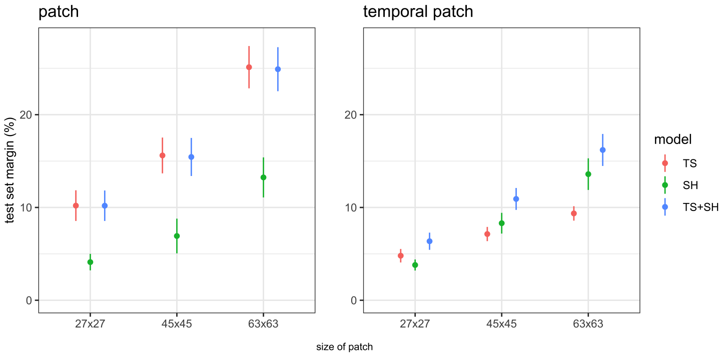


Figure: Random patch missing and temporal patch missing results. Three variants of our method are considered: TS, SH, TS+SH. The scatter points show the average test set RSE margin over baseline softImpute method, positive means performance better than softImpute. Error bar gives the 95% confidence interval.

Simulation Study: Imputation Example

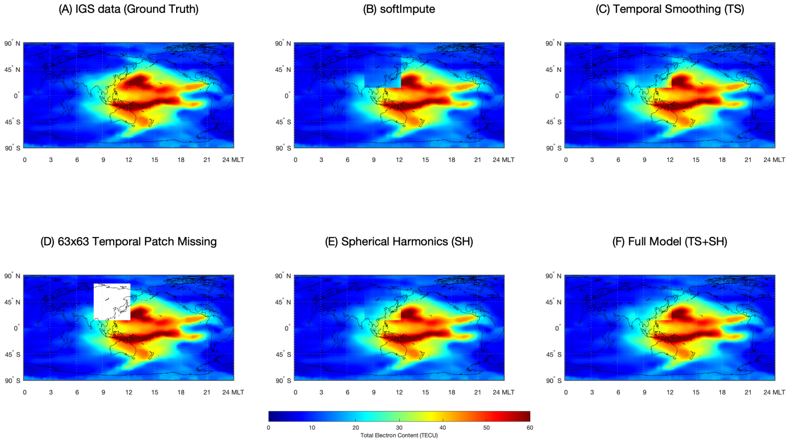


Figure: Example of imputing IGS data with temporal patch missingness.

Imputing Madrigal TEC map: Data

- To impute the final Madrigal TEC map, we fit VISTA on each day of TEC map, which is of size $181 \times 361 \times 288$. Every matrix is of size 181×361. We showcase our results based on two days of data: Sept-08-2017 (storm day), Sept-03-2017 (non storm day).

Imputing Madrigal TEC map: Data

- To impute the final Madrigal TEC map, we fit VISTA on each day of TEC map, which is of size $181 \times 361 \times 288$. Every matrix is of size 181×361. We showcase our results based on two days of data: Sept-08-2017 (storm day), Sept-03-2017 (non storm day).
- Tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ are determined with grid-search.

Imputing Madrigal TEC map: Data

- To impute the final Madrigal TEC map, we fit VISTA on each day of TEC map, which is of size $181 \times 361 \times 288$. Every matrix is of size 181×361. We showcase our results based on two days of data: Sept-08-2017 (storm day), Sept-03-2017 (non storm day).
- Tuning parameters $(\lambda_1, \lambda_2, \lambda_3)$ are determined with grid-search.
- Since Madrigal TEC map contains missing values, it not possible to directly validate the fitted model on the missing values. We instead randomly drop 20% of the observed pixels and use them as test set, and we fit our model only on the rest 80% of the observed pixels.

Imputing Madrigal TEC map: Result

Storm Day				
Model	test RSE	test MSE	# matrices better than softImpute	# matrices worse than Full model
softImpute ($\lambda_1 = 0.9$)	10.895%	2.675	/	285 (98.96%)
TS ($\lambda_1 = 0.9, \lambda_2 = 0.2$)	9.643%	2.106	284 (98.62%)	267 (92.71%)
SH ($\lambda_1 = 0.9, \lambda_3 = 0.021$)	9.936%	2.227	287 (99.65%)	274 (95.14%)
Full ($\lambda_1 = 0.9, \lambda_2 = 0.2, \lambda_3 = 0.021$)	9.357%	1.983	285 (98.96%)	/
Directly use Spherical Harmonics	17.354%	6.720	0 (0%)	288 (100%)
Non-Storm Day				
Model	test RSE	test MSE	# matrices better than softImpute	# matrices worse than Full model
softImpute ($\lambda_1 = 0.9$)	10.424%	1.324	/	283 (98.26%)
TS ($\lambda_1 = 0.9, \lambda_2 = 0.31$)	8.880%	0.958	281 (97.57%)	235 (81.60%)
SH ($\lambda_1 = 0.9, \lambda_3 = 0.03$)	9.231%	1.032	287 (99.65%)	278 (96.53%)
Full ($\lambda_1 = 0.9, \lambda_2 = 0.31, \lambda_3 = 0.03$)	8.592%	0.895	283 (98.26%)	/
Directly use Spherical Harmonics	15.732%	2.893	0 (0%)	288 (100%)

Table 1: Empirical study results from the madrigal database.

Figure: Imputation Result

Imputing Madrigal TEC map: Non-storm Day Example

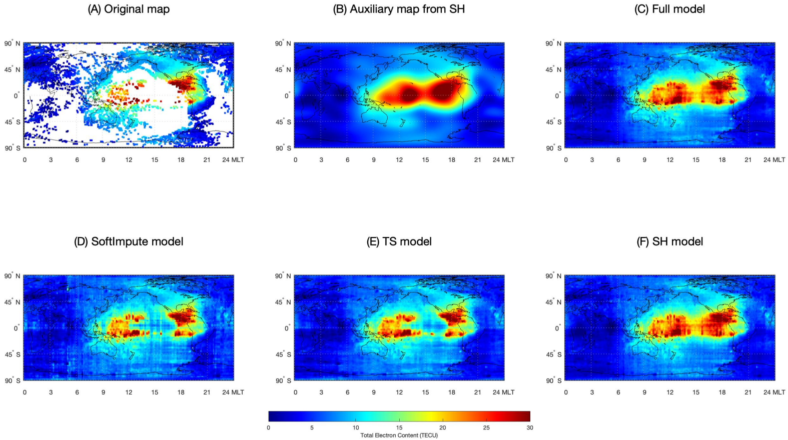


Figure: 2017-09-03/00:02:30 UT Result

Imputing Madrigal TEC map: Storm Day Example

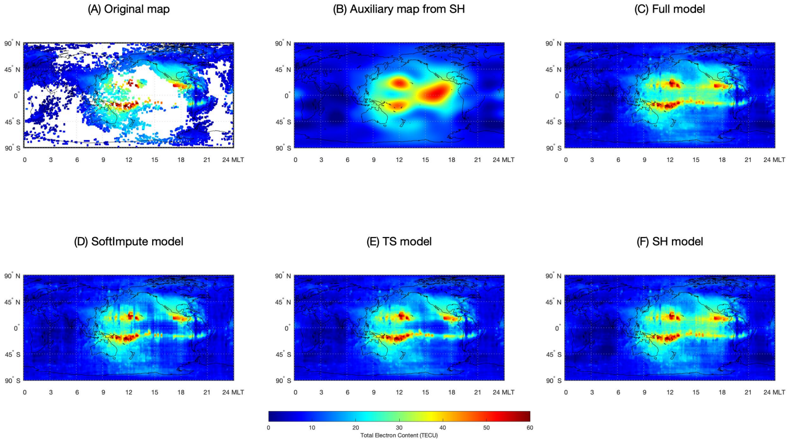


Figure: 2017-09-08/00:02:30 UT Result

Conclusion

- We propose a new imputation method (VISTA), combining matrix completion with soft rank constraint, temporal smoothing and spherical harmonics in a unified framework, to impute Total Electron Content (TEC) maps with over 50% data missing.

Conclusion

- We propose a new imputation method (VISTA), combining matrix completion with soft rank constraint, temporal smoothing and spherical harmonics in a unified framework, to impute Total Electron Content (TEC) maps with over 50% data missing.
- Matrix completion provides the basic low-rank structure of the imputation.

Conclusion

- We propose a new imputation method (VISTA), combining matrix completion with soft rank constraint, temporal smoothing and spherical harmonics in a unified framework, to impute Total Electron Content (TEC) maps with over 50% data missing.
- Matrix completion provides the basic low-rank structure of the imputation.
- Temporal smoothing borrows information from TEC maps at adjacent timestamp and smooth the low-rank structure.

Conclusion

- We propose a new imputation method (VISTA), combining matrix completion with soft rank constraint, temporal smoothing and spherical harmonics in a unified framework, to impute Total Electron Content (TEC) maps with over 50% data missing.
- Matrix completion provides the basic low-rank structure of the imputation.
- Temporal smoothing borrows information from TEC maps at adjacent timestamp and smooth the low-rank structure.
- Spherical harmonics provides a warm-start of imputation values at big patches of missingness.

Conclusion

- We propose a new imputation method (VISTA), combining matrix completion with soft rank constraint, temporal smoothing and spherical harmonics in a unified framework, to impute Total Electron Content (TEC) maps with over 50% data missing.
- Matrix completion provides the basic low-rank structure of the imputation.
- Temporal smoothing borrows information from TEC maps at adjacent timestamp and smooth the low-rank structure.
- Spherical harmonics provides a warm-start of imputation values at big patches of missingness.
- Empirical results suggest improvements on both global-scale and meso-scale reconstruction.

Future Plan

- We plan to release a data product containing the imputed TEC maps based on VISTA for the last solar cycle (2009-2020).

Future Plan

- We plan to release a data product containing the imputed TEC maps based on VISTA for the last solar cycle (2009-2020).
- We plan to use matrix/tensor-based factor model and other machine learning methods to do TEC map predictions using our VISTA data product as inputs. The ultimate goal is to provide a complete imputation-prediction pipeline for operational use.

References I

Candès, Emmanuel J and Terence Tao (2010). "The power of convex relaxation: Near-optimal matrix completion". In: *IEEE Transactions on Information Theory* 56.5, pp. 2053–2080.

Hastie, Trevor, Rahul Mazumder, Jason D Lee, and Reza Zadeh (2015). "Matrix completion and low-rank SVD via fast alternating least squares". In: *The Journal of Machine Learning Research* 16.1, pp. 3367–3402.

Mazumder, Rahul, Trevor Hastie, and Robert Tibshirani (2010). "Spectral regularization algorithms for learning large incomplete matrices". In: *The Journal of Machine Learning Research* 11, pp. 2287–2322.

Nortje, Caitlin R, Wil OC Ward, Bartosz P Neuman, and Li Bai (2015). "Spherical harmonics for surface parametrisation and remeshing". In: *Mathematical Problems in Engineering* 2015.

References II

Pan, Yang, Mingwu Jin, Shunrong Zhang, and Yue Deng (2020). "TEC map completion using DCGAN and Poisson blending". In: *Space Weather* 18.5, e2019SW002390.

Srebro, Nathan, Jason Rennie, and Tommi S Jaakkola (2005). "Maximum-margin matrix factorization". In: *Advances in neural information processing systems*, pp. 1329–1336.