

# Algoritmi za reševanje problema matričnih napolnitev

Matej Klančar

Mentor: doc. dr. Aljaž Zalar

2023

1 Predstavitev problema

2 Algoritmi

3 Rezultati

4 Zaključek

# Predstavitev problema

- Problem sprejme vhodno matriko  $M$
- Nekaterih elementov **ne poznamo**
- Kako določiti manjkajoče elemente, da bo **rang čim manjši**

$$\arg \min_X \text{rank}(X)$$

pod pogojem  $\mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M)$

# Uporabe algoritmov

- Priporočilni sistemi
- Rekonstrukcija signalov
- Računanje razdalj med napravami
- Rekonstrukcija slik

# Kazalo

1 Predstavitev problema

2 Algoritmi

3 Rezultati

4 Zaključek

# Algoritem NNM

- Minimizira nuklearno normo

$$\|A\|_* = \sum_{i=1}^n \sigma_i(A)$$

- Rešuje problem semidefinitega programiranja
- Lahko rešujemo s primernimi orodji za reševanje takih problemov (npr. Sedumi)

# Algoritem SVT

- Problem definira z Lagrangeovo funkcijo

$$\mathcal{L}(X, Y) = \tau \|X\|_* + \frac{1}{2} \|X\|_F^2 + \langle Y, \mathcal{P}_\Omega(M - X) \rangle$$

- Uporabimo t.i. Uzawa algoritem, rešujemo iterativno

$$X^{(k)} = \mathcal{D}_\tau(Y^{(k-1)}),$$

$$Y^{(k)} = Y^{(k-1)} + \delta_k \mathcal{P}_\Omega(M - X^{(k)}),$$

- Operator praga  $\mathcal{D}_\tau : \mathbb{R}^{n_1 \times n_2} \rightarrow \mathbb{R}^{n_1 \times n_2}$  je definiran kot

$$\mathcal{D}_\tau(A) := U \mathcal{D}_\tau(\Sigma) V^T,$$

$$\begin{aligned} \mathcal{D}_\tau(\Sigma) = \text{diag} \left( \max(\sigma_1 - \tau, 0), \max(\sigma_2 - \tau, 0), \dots, \right. \\ \left. \max(\sigma_{\min(n_1, n_2)} - \tau, 0) \right) \end{aligned}$$

- Uporabi idejo, da ima matrika majhnega ranga le nekaj velikih singularnih vrednosti.

# Algoritem TNNM

- Minimizira pritezano nuklearno normo

$$\|X\|_r = \sum_{i=r+1}^{\min(n_1, n_2)} \sigma_i(X)$$

- Uporabi  $A_I, B_I$ , sestavljeni iz singularnih vektorjev  $X$
- Uporablja iterativni algoritem ADMM, definira pomožni matriki  $W, Y$

$$X^{(k+1)} = \mathcal{D}_{\frac{1}{\beta}}(W^{(k)} - \frac{1}{\beta}Y^{(k)})$$

$$W^{(k+1)} = X^{(k+1)} + \frac{1}{\beta}(A_I^T B_I + Y^{(k)})$$

$$Y^{(k+1)} = Y^{(k)} + \beta(X^{(k+1)} - W^{(k+1)})$$

# Algoritem ASD

- Išče matriki  $X \in \mathbb{R}^{n_1 \times r}$ ,  $Y \in \mathbb{R}^{r \times n_2}$  katerih produkt bo enak rekonstrukciji
- Uporablja gradientni spust nad funkcijo

$$f(X, Y) = \frac{1}{2} \|\mathcal{P}_\Omega(M) - \mathcal{P}_\Omega(XY)\|_F^2$$

- Z uporabo gradienta in optimalnega koraka izvaja premik

$$\begin{aligned} X^{(k+1)} &= X^{(k)} - t_{x^{(k)}} \nabla f_{Y^{(k)}}(X^{(k)}) \\ Y^{(k+1)} &= Y^{(k)} - t_{y^{(k)}} \nabla f_{X^{(k+1)}}(Y^{(k)}) \end{aligned}$$

# Algoritem LMaFit

- Uporablja Moore-Penrose inverz
- Množenje rešuje po metodi najmanjših kvadratov

$$X^{(k+1)} = Z^{(k)}(Y^{(k)})^\dagger$$

$$Y^{(k+1)} = (X^{(k+1)})^\dagger Z^{(k)}$$

$$Z^{(k+1)} = X^{(k+1)}Y^{(k+1)} + \mathcal{P}_\Omega(M - X^{(k+1)}Y^{(k+1)})$$

# Kazalo

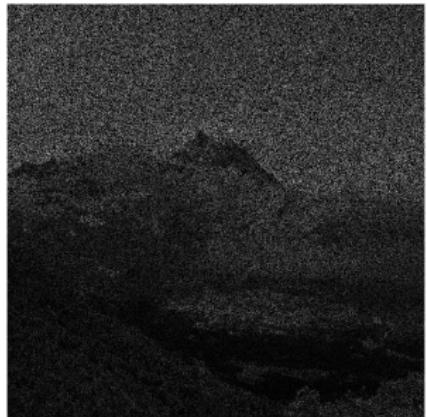
1 Predstavitev problema

2 Algoritmi

3 Rezultati

4 Zaključek

# Vhodne slike



(a) 35% znanih podatkov

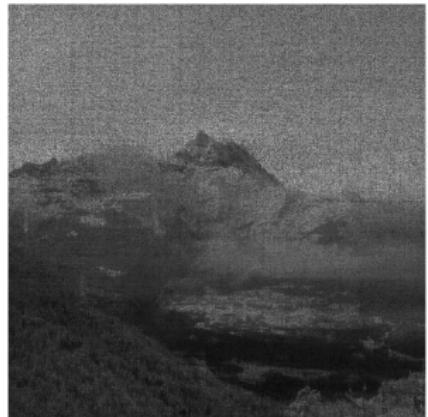


(b) 45% znanih podatkov



(c) 60% znanih podatkov

# Rekonstrukcija z algoritmom SVT



(a) 35% znanih podatkov



(b) 45% znanih podatkov



(c) 60% znanih podatkov

# Rekonstrukcija z algoritmom TNNM



(a) 35% znanih podatkov

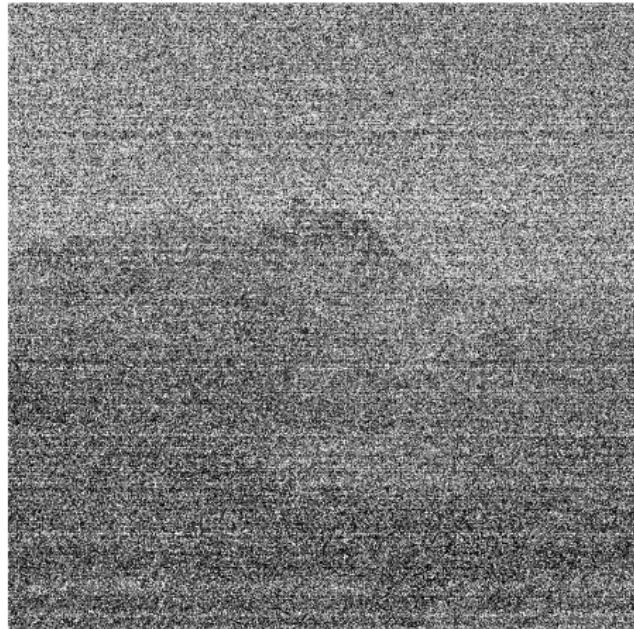


(b) 45% znanih podatkov



(c) 60% znanih podatkov

# Rekonstrukcija z algoritmom ASD



(a) 35% znanih podatkov



(b) 45% znanih podatkov

# Rekonstrukcija z algoritmom LMaFit



(a) 35% znanih podatkov

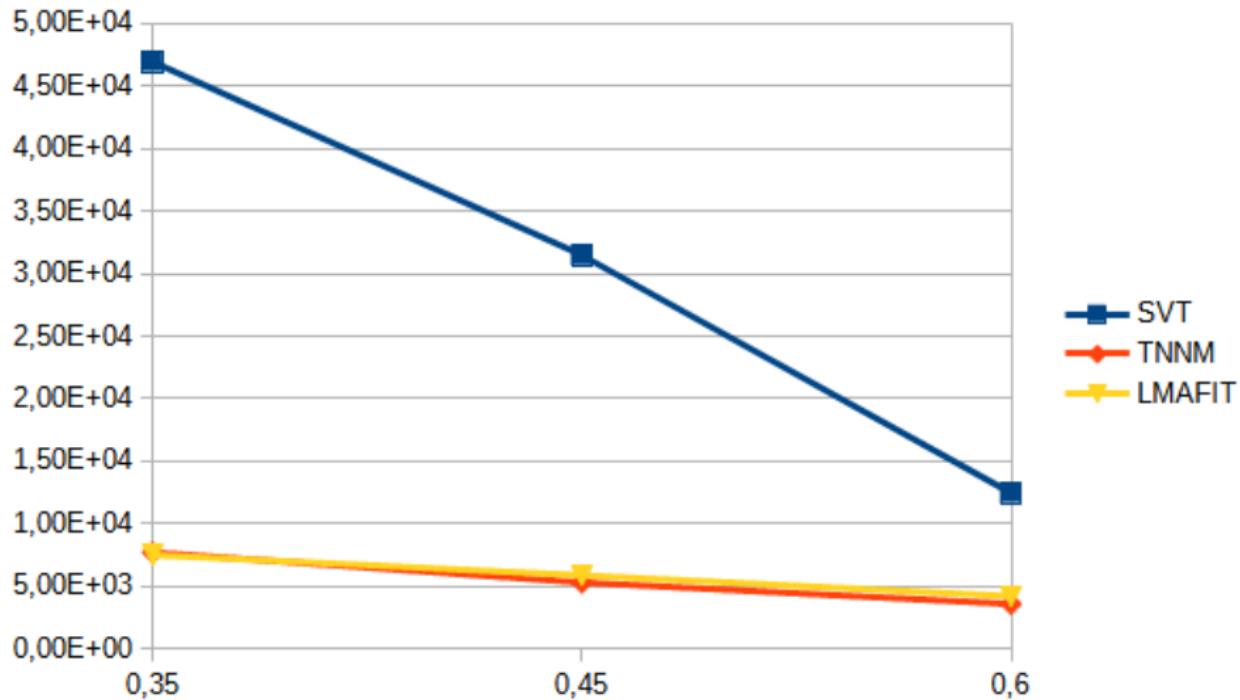


(b) 45% znanih podatkov

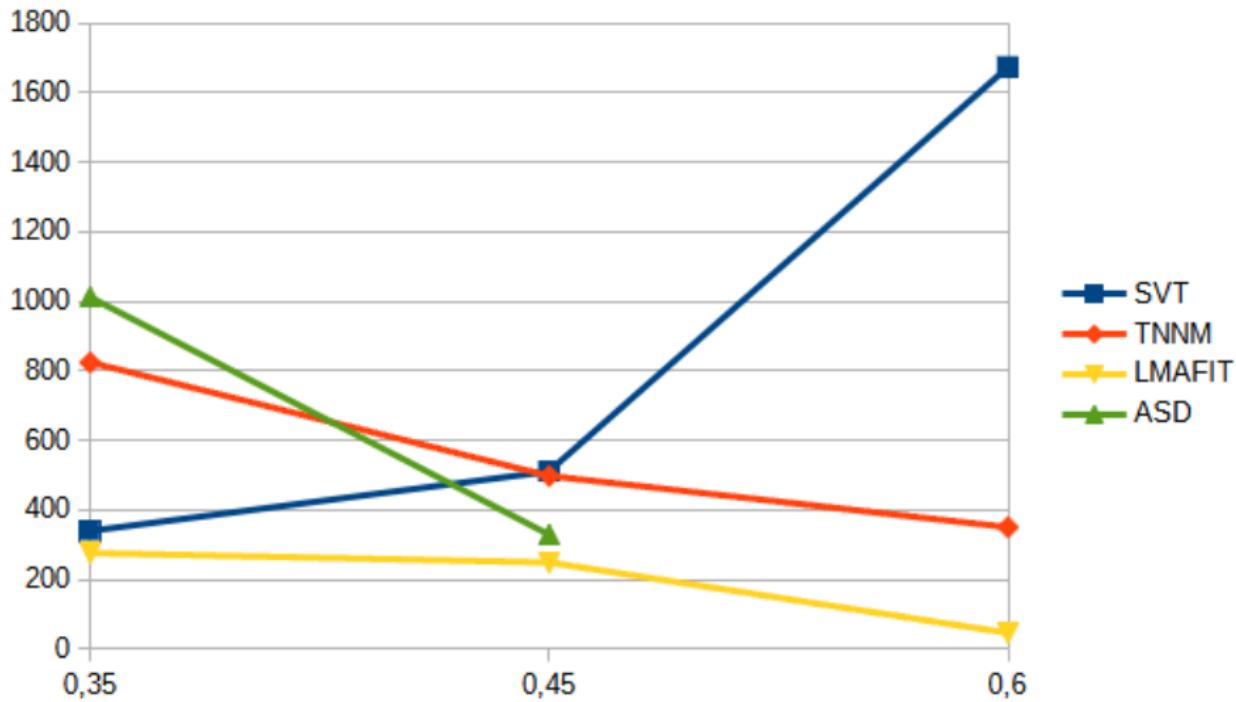


(c) 60% znanih podatkov

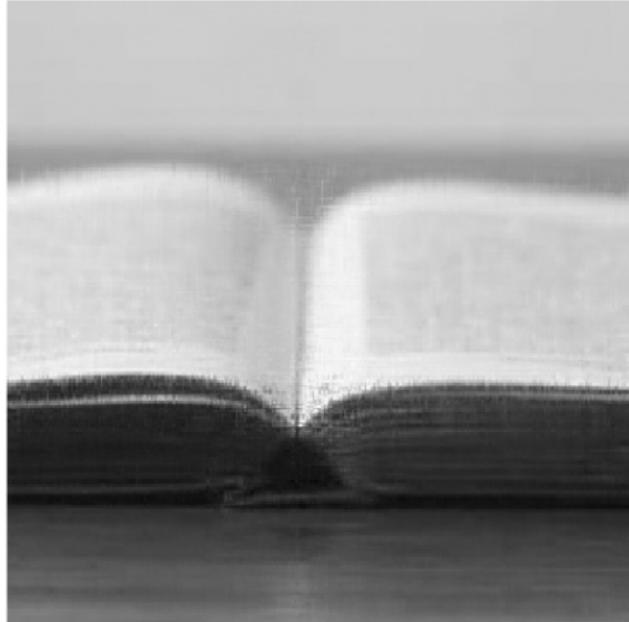
# Napake algoritmov v Frobeniusovi normi



# Časi izvajanja algoritmov



# Vpliv kompleksnosti motiva pri algoritmu TNNM



(a) 35% znanih podatkov

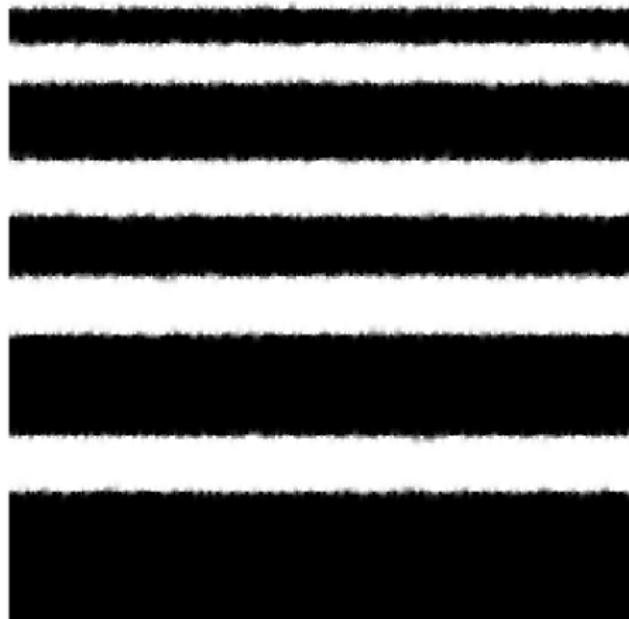


(b) 35% znanih podatkov

# Rekonstrukcija z reševanjem Laplaceovih diferencialnih enačb



(a) 35% znanih podatkov



(b) 35% znanih podatkov

# Kazalo

1 Predstavitev problema

2 Algoritmi

3 Rezultati

4 Zaključek

# Ugotovitve

- Algoritem LMaFit je najhitrejši
- Algoritem TNNM vrne najboljše rezultate
- Prednost algoritma SVT - ne potrebujemo informacije o rangu
- Algoritem NNM primeren za majhne matrike ( $100 \times 100$ )
- Ker se pri matričnih napolnitvah ne smemo zanašati na lokalno podobnost, obstajajo za rekonstrukcijo slik boljše metode

# Glavni prispevki

- Implementacija algoritmov
- Primerjava delovanja algoritmov
- Analiza vplivov parametrov
- Interpretacija rezultatov prek matematičnega ozadja