UNIVERSITY OF CAMBRIDGE
Microsoft
Je Hyeong Hong
Andrew W. Fitzgibbon

Secrets of Matrix Factorization:

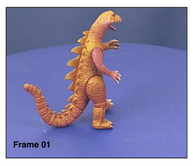
1. Matrix factorization with missing data

Given a noisy measurement matrix $\mathrm{M} \in \mathbb{R}^{m\times n}$, where some of the values are unobserved or weighted by another matrix $\mathrm{W} \in \mathbb{R}^{m\times n}$, we are interested in finding a set of low-rank matrices $\mathrm{U} \in \mathbb{R}^{m\times r}$ and $\mathrm{V} \in \mathbb{R}^{n\times r}$ that minimizes the error function

$$f(\mathrm{U},\mathrm{V}) = \|\boldsymbol{\varepsilon}(\mathrm{U},\mathrm{V})\|_2^2 = \|\mathrm{W} \odot (\mathrm{U}\mathrm{V}^\top - \mathrm{M})\|_F^2 + \mu(\|\mathrm{U}\|_F^2 + \|\mathrm{V}\|_F^2) \qquad (1.1)$$

where the operator $\odot$ is Hadamard or element-wise product, and the norms are Frobenius.

- This type of low-rank models appears in rigid and non-rigid structure-from-motion (SfM), photometric stereo and recommender systems.
- For computer vision problems, the regularization parameter μ is often set to 0.



(a) Rigid SfM [3]


(b) Non-rigid SfM [3]


(c) Photometric Stereo [4]

2. Known strategies in the literature

A. Block coordinate-descent (ALS)

- Fix U and optimize over V, and vice versa.
- Closed form updates for both U and V.

inputs: M, W, r, U, μ
$\widetilde{\mathrm{W}} \leftarrow \mathrm{diag}\big(\mathrm{vec}(\mathrm{W})\big)$ with zero-rows removed.
$\widetilde{\mathbf{m}} \leftarrow \widetilde{\mathrm{W}}\,\mathrm{vec}(\mathrm{M})$
repeat
  $\mathrm{U} \leftarrow \mathrm{unvec}\big(\widetilde{\mathrm{V}}^{-\mu}\widetilde{\mathbf{m}}\big)$
  $\mathrm{V} \leftarrow \mathrm{unvec}\big(\widetilde{\mathrm{U}}^{-\mu}\widetilde{\mathbf{m}}\big)$
until convergence
outputs: U, V

- $\mathrm{X}^{-\mu} \coloneqq (\mathrm{X}^\top\mathrm{X} + \mu\mathrm{I})^{-1}\mathrm{X}^\top$ (When $\mu = 0$, $\mathrm{X}^{-\mu} = \mathrm{X}^\dagger$.)
- $\widetilde{\mathrm{U}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{I} \otimes \mathrm{U})$
- $\widetilde{\mathrm{V}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{V} \otimes \mathrm{I})$
- $\mathrm{V}^*(\mathrm{U}) = \mathrm{unvec}\big(\widetilde{\mathrm{U}}^{-\mu}\widetilde{\mathbf{m}}\big)$

B. Joint optimization [3]

- Optimize over U and V simultaneously using a Newton-like solver.
- Bilinear structure → sparse Hessian (or JTJ) matrix.

inputs: M, W, r, U, V, μ, λ_0
$\lambda \leftarrow \lambda_0$
$\widetilde{\mathrm{W}} \leftarrow \mathrm{diag}\big(\mathrm{vec}(\mathrm{W})\big)$ with zero-rows removed.
$\widetilde{\mathbf{m}} \leftarrow \widetilde{\mathrm{W}}\,\mathrm{vec}(\mathrm{M})$
repeat
  $\mathbf{z} \leftarrow [\mathrm{vec}(\mathrm{U}) \quad ; \quad \mathrm{vec}(\mathrm{V}^\top)]$
  Compute the gradient $\mathbf{g} \coloneqq df(\mathrm{U},\mathrm{V})/d\mathbf{z}$.
  Compute the Hessian $\mathrm{H} \coloneqq d^2 f(\mathrm{U},\mathrm{V})/d\mathbf{z}^2$ or its approximation.
  repeat
    $\Delta\mathbf{z} \leftarrow (\mathrm{H} + \lambda\mathrm{I})^{-1}\mathbf{g}$
    Retrieve $\Delta\mathrm{U}$ and $\Delta\mathrm{V}$ from $\Delta\mathbf{z}$.
    $\lambda \leftarrow \lambda * 10$
  until $f(\mathrm{U}+\Delta\mathrm{U},\mathrm{V}+\Delta\mathrm{V}) < f(\mathrm{U},\mathrm{V})$
  $\mathrm{U} \leftarrow \mathrm{U} + \Delta\mathrm{U}$
  $\mathrm{V} \leftarrow \mathrm{V} + \Delta\mathrm{V}$
  $\lambda \leftarrow \lambda/100$
until convergence
outputs: U, V

C. Variable projection (VarPro) [6]

- Replace V by its closed form minimizer $\mathrm{V}^*(\mathrm{U}) \coloneqq \mathrm{argmin}_\mathrm{V} f(\mathrm{U},\mathrm{V}) = \mathrm{unvec}\big(\widetilde{\mathrm{U}}^{-\mu}\widetilde{\mathbf{m}}\big)$.
- Optimize the reduced problem over U using a Newton-like solver.
- Nonlinear structure → denser Hessian (or JTJ) matrix.

inputs: M, W, r, U, μ, λ_0
$\lambda \leftarrow \lambda_0$
$\widetilde{\mathrm{W}} \leftarrow \mathrm{diag}\big(\mathrm{vec}(\mathrm{W})\big)$ with zero-rows removed.
$\widetilde{\mathbf{m}} \leftarrow \widetilde{\mathrm{W}}\,\mathrm{vec}(\mathrm{M})$
repeat
  Compute the gradient $\mathbf{g} \coloneqq df(\mathrm{U},\mathrm{V}^*(\mathrm{U}))/d\mathrm{vec}(\mathrm{U})$.
  Compute the Hessian $\mathrm{H} \coloneqq d^2 f(\mathrm{U},\mathrm{V}^*(\mathrm{U}))/d\mathrm{vec}(\mathrm{U})^2$ or its approximation.
  repeat
    $\Delta\mathrm{U} \leftarrow \mathrm{unvec}\big((\mathrm{H} + \lambda\mathrm{I})^{-1}\mathbf{g}\big)$
    $\lambda \leftarrow \lambda * 10$
  until $f\big(\mathrm{U}+\Delta\mathrm{U},\mathrm{V}^*(\mathrm{U}+\Delta\mathrm{U})\big) < f(\mathrm{U},\mathrm{V}^*(\mathrm{U}))$
  $\mathrm{U} \leftarrow \mathrm{U} + \Delta\mathrm{U}$
  $\mathrm{V} \leftarrow \mathrm{V}^*(\mathrm{U})$
  $\lambda \leftarrow \lambda/100$
until convergence
outputs: U, V

- $\mathrm{X}^{-\mu} \coloneqq (\mathrm{X}^\top\mathrm{X} + \mu\mathrm{I})^{-1}\mathrm{X}^\top$ (When $\mu = 0$, $\mathrm{X}^{-\mu} = \mathrm{X}^\dagger$.)
- $\widetilde{\mathrm{U}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{I} \otimes \mathrm{U})$
- $\widetilde{\mathrm{V}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{V} \otimes \mathrm{I})$
- $\mathrm{V}^*(\mathrm{U}) = \mathrm{unvec}\big(\widetilde{\mathrm{U}}^{-\mu}\widetilde{\mathbf{m}}\big)$

3. Random restarts

- Any algorithm started from a random starting point (i.e. random U and/or V) is improved by starting from multiple points.
- To find a strong optimum, just run from many random starting points. Report optimum that is seen twice and is lower than all the others. Hence,

A new meta algorithm, RUSSO-*X*

RUSSO-*X* is literally "run algorithm-*X* until seen second optimum", where this optimum is the best one observed so far. It is a simple meta-algorithm which wraps any existing low-rank matrix factorization algorithm and improves its success rate.

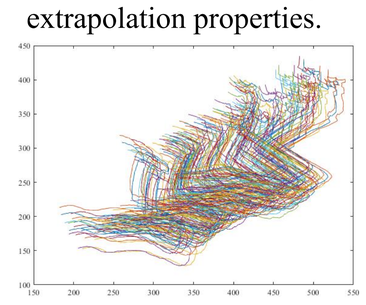
inputs: M, W, r, X, N
$k \leftarrow 0$
$f_{best} \leftarrow 0$
repeat
  $\mathrm{U} \leftarrow \mathrm{randn}(m, r)$ // m is the row size of M.
  $f_{current} \leftarrow \min_{\mathrm{U},\mathrm{V}} f(\mathrm{U},\mathrm{V})$ using algorithm-X
  if $|f_{current} - f_{best}| < 10^{-6}$ **do**
    break // output the RUSSO-X min. value.
  else if $f_{current} < f_{best}$ **do**
    $f_{best} \leftarrow f_{current}$ // update the current candidate for RUSSO-X min.
  end
  $k \leftarrow k + 1$
until $k = N$
output: f_{best}

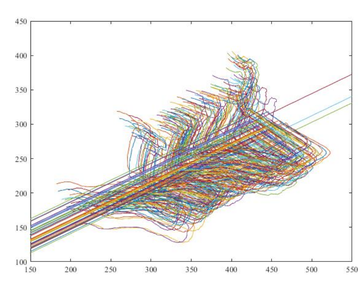
- An example of 10 runs from random starting points illustrates how RUSSO-*X* works in practice. (NB: RUSSO-*X* still returns the best optimum observed so far even when it is not observed twice.)

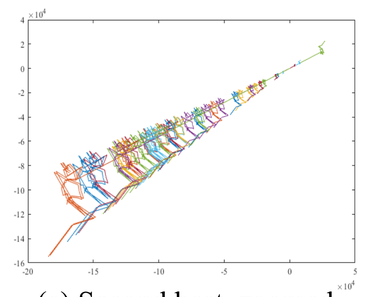
Run sequence	1	2	3	4	5	6	7	8	9	10	Success rate
Minimum	1.42	1.58	1.42	1.14	1.31	**1.02**	2.04	**1.02**	**1.02**	1.28	3 / 10
RUSSO-*X* min.	1.42	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.28	8 / 10

A side note: Why do we care about the best optimum?

The reconstruction of point trajectories (GIR in §8) shown below illustrates that a solution with function value just .06% above the optimum can have significantly worse extrapolation properties.


(a) Best known minimum (0.3228)


(b) Second best solution (0.3230)


(c) Second best, zoomed to image

4. Approximations

- All top matrix factorization algorithms in computer vision [5, 7, 9] benefit from the use of the **unregularized** VarPro, which is illustrated in §2.C.
- These just differ in how much approximation each makes in computing the Hessian. We use the RW*Y* convention where *Y* is the algorithm number mentioned in [10].

$$\frac{1}{2}\mathrm{H} = \underbrace{\widetilde{\mathrm{V}}^{*\top}\widetilde{\mathrm{V}}^*}_{0.5\,\mathrm{H}_{RW3}} - \underbrace{\widetilde{\mathrm{V}}^{*\top}\widetilde{\mathrm{U}}\widetilde{\mathrm{U}}^\dagger\widetilde{\mathrm{V}}^*}_{\mathrm{G}_{RW2}} - \underbrace{\mathrm{K}_{mr}^\top\mathrm{Z}^*\big(\widetilde{\mathrm{U}}^\top\widetilde{\mathrm{U}}\big)^{-1}\mathrm{Z}^{*\top}\mathrm{K}_{mr}}_{\mathrm{G}_{RW1}} + \underbrace{\mathrm{K}_{mr}^\top\mathrm{Z}^*\widetilde{\mathrm{U}}^\dagger\widetilde{\mathrm{V}}^*}_{\mathrm{G}_{FN}} + \underbrace{\widetilde{\mathrm{V}}^{*\top}\widetilde{\mathrm{U}}^{\dagger\top}\mathrm{Z}^{*\top}\mathrm{K}_{mr}}_{\mathrm{G}_{FN}^\top}$$

$$\frac{1}{2}\mathrm{H}_{RW2} = \frac{1}{2}\mathrm{H}_{RW3} - \mathrm{G}_{RW2}$$
$$\frac{1}{2}\mathrm{H}_{RW1} = \frac{1}{2}\mathrm{H}_{RW2} + \mathrm{G}_{RW1}$$
$$\frac{1}{2}\mathrm{H} = \frac{1}{2}\mathrm{H}_{RW2} - \mathrm{G}_{RW1} + \mathrm{G}_{FN} + \mathrm{G}_{FN}^\top$$

- $\mathrm{K}_{mr}\mathrm{vec}(\mathrm{U}) = \mathrm{vec}(\mathrm{U}^\top)$
- $\widetilde{\mathrm{U}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{I} \otimes \mathrm{U})$
- $\mathrm{V}^*(\mathrm{U}) = \mathrm{unvec}\big(\widetilde{\mathrm{U}}^{-\mu}\widetilde{\mathbf{m}}\big)$
- $\widetilde{\mathrm{V}}^*(\mathrm{U}) \coloneqq \widetilde{\mathrm{W}}(\mathrm{V}^*(\mathrm{U}) \otimes \mathrm{I})$
- $\mathrm{Z}^*(\mathrm{U}) \coloneqq \big(\mathrm{W} \odot \mathrm{W} \odot (\mathrm{M} - \mathrm{U}\mathrm{V}^{*\top}(\mathrm{U}))\big) \otimes \mathrm{I}$

- The most influential term is $-\widetilde{\mathrm{V}}^*\widetilde{\mathrm{U}}\widetilde{\mathrm{U}}^\dagger\widetilde{\mathrm{V}}^*$, which brings accuracy from $O(0)$ to $O(100)\%$.

5. Manifold optimization

- As noted in [3], the unregularized matrix factorization problem have an innate gauge freedom. ($\mathrm{U}\mathrm{V}^\top = (\mathrm{U}\mathrm{A}^{-1})(\mathrm{V}\mathrm{A}^\top)^\top$ for any invertible matrix A.)
- For unregularized VarPro, this translates to U residing in the Grassmann manifold. i.e. U is a solution → entire $\mathrm{col}(\mathrm{U})$ is a solution space.
- To incorporate the manifold structure, we need to include the following steps in addition to the pseudocode illustrated in §2.C. (Background theory in [1].)

At each iteration,
1. $\mathbf{g} \leftarrow \mathrm{P}^\top\mathbf{g}$ // project the gradient and the Hessian to the tangent space of U.
2. $\mathrm{H} \leftarrow \mathrm{P}\mathrm{H}\mathrm{P}^\top$ // project the Hessian to the tangent space of U.
3. $\mathrm{U} \leftarrow \mathrm{unvec}\big(qf(\mathrm{U}+\Delta\mathrm{U})\big)$ // retract updated U back to the Grassmann manifold.

NB: For this problem, P can be $\mathrm{I} \otimes \mathrm{U}_\perp^\top \in \mathbb{R}^{(m-r)r\times mr}$ or $\mathrm{I} \otimes (I - \mathrm{U}\mathrm{U}^\top) \in \mathbb{R}^{mr\times mr}$.

Acknowledgement The work was supported by Microsoft and Toshiba Research Europe. We thank Roberto Cipolla, Christopher Zach, Bamdev Mishra and the anonymous reviewers.
The conference travel was supported by the British Machine Vision Association (BMVA), the Cambridge University Engineering Department (Rex Moir Fund), Christ's College (University of Cambridge) and the Cambridge Philosophical Society.

Project repository https://github.com/jhh37/matrix-factorization/

Approximations, Numerics, Manifold Optimization and Random Restarts

6. Unified analysis of algorithms

A. Alternation and VarPro algorithms for the unregularized problem ($\mu \leftarrow 0$)

ALS	VH_PF [8]	TW_WB [11]	TO_DW [9]	DRW1	DRW1P	DRW2	DRW2P	PG_CSF [7]	CH_LM_S [5]	CH_LM_S_GN [5]	CH_LM_S_RW2	CH_LM_M [5]	CH_LM_M_GN [5]	CH_LM_M_RW2	NB_RTRMC [2]	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1:
		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2:
				3	3					3			3			3:
									4			4			4	4:
					(5)		(5)								(5)	5:
												6	6	6		6:
					(7)		(7)					7	7	7	(7)	7:
					(8)		(8)					8	8	8	(8)	8:
			9		9		9									9:
		10	10	10	10	10	10	10	10	10	10	10	10	10	10	10:
11	11	11														11:
			12	12	12	12	12	12	12	12	12	12	12	12	12	12:
			13	13	13	13	13	13	13	13	13	13	13	13	13	13:
		14	14	14	14	14	14	14	14	14	14	14	14	14	14	14:
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15:
	16				16											16:
17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17:
			18	18	18	18	18	18	18	18	18	18	18	18	18	18:

inputs: M, W, r, U, λ_0
$\lambda \leftarrow \lambda_0$
$\widetilde{\mathrm{W}} \leftarrow \mathrm{diag}\big(\mathrm{vec}(\mathrm{W})\big)$ with zero-rows removed.
$\widetilde{\mathbf{m}} \leftarrow \widetilde{\mathrm{W}}\,\mathrm{vec}(\mathrm{M})$
repeat
  $\mathbf{g} \leftarrow \mathrm{V}^*(\mathrm{U})^\top \mathrm{vec}(\mathrm{U}\mathrm{V}^*(\mathrm{U})^\top - \mathrm{M})$ // for all algorithms listed here.
  $\mathrm{H} \leftarrow \mathrm{V}^{*\top}\mathrm{V}^*$
  $\mathrm{H} \leftarrow \mathrm{H} - \widetilde{\mathrm{V}}^{*\top}\widetilde{\mathrm{U}}\widetilde{\mathrm{U}}^\dagger\widetilde{\mathrm{V}}^*$
  $\mathrm{H} \leftarrow \mathrm{H} + \mathrm{K}_{mr}^\top \mathrm{Z}^*\big(\widetilde{\mathrm{U}}^\top\widetilde{\mathrm{U}}\big)^{-1}\mathrm{Z}^{*\top}\mathrm{K}_{mr}$
  $\mathrm{H} \leftarrow \mathrm{H} - \mathrm{K}_{mr}^\top \mathrm{Z}^*\big(\widetilde{\mathrm{U}}^\top\widetilde{\mathrm{U}}\big)^{-1}\mathrm{Z}^{*\top}\mathrm{K}_{mr} + \mathrm{K}_{mr}^\top \mathrm{Z}^*\widetilde{\mathrm{U}}^\dagger\widetilde{\mathrm{V}}^* + \widetilde{\mathrm{V}}^{*\top}\widetilde{\mathrm{U}}^{\dagger\top}\mathrm{Z}^{*\top}\mathrm{K}_{mr}$
  $\mathrm{P} \leftarrow \mathrm{I} \otimes (\mathrm{I} - \mathrm{U}\mathrm{U}^\top) \in \mathbb{R}^{mr\times mr}$ // (5) is also a no-operation if 7 and 8 are.
  $\mathrm{P} \leftarrow \mathrm{I} \otimes \mathrm{U}_\perp^\top \in \mathbb{R}^{(m-r)r\times mr}$
  $\mathbf{g} \leftarrow \mathrm{P}\mathbf{g}$ // (7) is a no-operation since $\mathbf{g} = \mathrm{P}\mathbf{g}$.
  $\mathrm{H} \leftarrow \mathrm{P}\mathrm{H}\mathrm{P}^\top$ // (8) is a no-operation since $\mathrm{H} = \mathrm{P}\mathrm{H}$.
  $\mathrm{H} \leftarrow \mathrm{H} + \mathrm{I} \otimes \mathrm{U}\mathrm{U}^\top$ // relaxed constraint to promote $\mathrm{U}^\top\Delta\mathrm{U} = 0$.
  repeat
    $\Delta\mathrm{U} \leftarrow \mathrm{unvec}(\mathrm{H}^{-1}\mathbf{g})$
    $\Delta\mathrm{U} \leftarrow \mathrm{unvec}\big((\mathrm{H} + \lambda\mathrm{I})^{-1}\mathbf{g}\big)$
    $\lambda \leftarrow \lambda * 10$
  until $f\big(\mathrm{U} + \Delta\mathrm{U}, \mathrm{V}^*(\mathrm{U} + \Delta\mathrm{U})\big) < f(\mathrm{U}, \mathrm{V}^*(\mathrm{U}))$
  $\mathrm{U} \leftarrow \mathrm{U} + \Delta\mathrm{U}$
  $\mathrm{U} \leftarrow qf(\mathrm{U})$ // [U,~]=qr(U,0) in MATLAB.
  $\mathrm{V} \leftarrow \mathrm{unvec}\big(\widetilde{\mathrm{U}}^\dagger\widetilde{\mathbf{m}}\big)$
  $\lambda \leftarrow \lambda/100$
until convergence
outputs: U, V

- $\mathrm{K}_{mr}\mathrm{vec}(\mathrm{U}) = \mathrm{vec}(\mathrm{U}^\top)$
- $\widetilde{\mathrm{U}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{I} \otimes \mathrm{U})$
- $\mathrm{V}^*(\mathrm{U}) = \mathrm{unvec}\big(\widetilde{\mathrm{U}}^\dagger\widetilde{\mathbf{m}}\big)$
- $\widetilde{\mathrm{V}}^*(\mathrm{U}) \coloneqq \widetilde{\mathrm{W}}(\mathrm{V}^*(\mathrm{U}) \otimes \mathrm{I})$
- $\mathrm{Z}^*(\mathrm{U}) \coloneqq \big(\mathrm{W} \odot \mathrm{W} \odot (\mathrm{M} - \mathrm{U}\mathrm{V}^{*\top}(\mathrm{U}))\big) \otimes \mathrm{I}$

B. Joint optimization algorithms

CE_LM	AB_DN [3]	
1	1	1:
	2	2:

inputs: M, W, r, U, V, μ, λ_0
$\lambda \leftarrow \lambda_0$
$\widetilde{\mathrm{W}} \leftarrow \mathrm{diag}\big(\mathrm{vec}(\mathrm{W})\big)$ with zero-rows removed.
$\widetilde{\mathbf{m}} \leftarrow \widetilde{\mathrm{W}}\,\mathrm{vec}(\mathrm{M})$
repeat
  $\mathbf{g} \leftarrow \begin{bmatrix} \widetilde{\mathrm{V}}^\top\mathrm{vec}(\mathrm{U}\mathrm{V}^\top - \mathrm{M}) + \mu\mathrm{vec}(\mathrm{U}) \\ \widetilde{\mathrm{U}}^\top\mathrm{vec}(\mathrm{U}\mathrm{V}^\top - \mathrm{M}) + \mu\mathrm{vec}(\mathrm{V}^\top) \end{bmatrix}$
  $\mathrm{H} \leftarrow \begin{bmatrix} \widetilde{\mathrm{V}}^\top\widetilde{\mathrm{V}} + \mu\mathrm{I} & \widetilde{\mathrm{V}}^\top\widetilde{\mathrm{U}} \\ \widetilde{\mathrm{U}}^\top\widetilde{\mathrm{V}} & \widetilde{\mathrm{U}}^\top\widetilde{\mathrm{U}} + \mu\mathrm{I} \end{bmatrix}$
  $\mathrm{H} \leftarrow \mathrm{H} + \begin{bmatrix} 0 & \mathrm{K}_{mr}^\top Z \\ \mathrm{Z}^\top\mathrm{K}_{mr} & 0 \end{bmatrix}$
  repeat
    $\Delta\mathbf{z} \leftarrow (\mathrm{H} + \lambda\mathrm{I})^{-1}\mathbf{g}$
    Retrieve ΔU and ΔV from $\Delta\mathbf{z}$.
    $\lambda \leftarrow \lambda * 10$
  until $f(\mathrm{U} + \Delta\mathrm{U}, \mathrm{V} + \Delta\mathrm{V}) < f(\mathrm{U}, \mathrm{V})$
  $\mathrm{U} \leftarrow \mathrm{U} + \Delta\mathrm{U}$
  $\mathrm{V} \leftarrow \mathrm{V} + \Delta\mathrm{V}$
  $\lambda \leftarrow \lambda/100$
until convergence
outputs: U, V

- $\mathrm{K}_{mr}\mathrm{vec}(\mathrm{U}) = \mathrm{vec}(\mathrm{U}^\top)$
- $\widetilde{\mathrm{U}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{I} \otimes \mathrm{U})$
- $\widetilde{\mathrm{V}} \coloneqq \widetilde{\mathrm{W}}(\mathrm{V} \otimes \mathrm{I})$
- $\mathrm{Z} \coloneqq (\mathrm{W} \odot \mathrm{W} \odot (\mathrm{M} - \mathrm{U}\mathrm{V}^\top) \otimes \mathrm{I}$

7. Numerics

A. Numerical issues

- Mainly, computation of the Hessian (or its approximation) and its inversion.
- Use of the QR factorization ($\widetilde{\mathrm{U}} = \widetilde{\mathrm{U}}_Q\widetilde{\mathrm{U}}_R$) improves accuracy and performance.
- Cholesky decomposition is used for computing ΔU.

B. Profiler-guided optimization

- Standard Matlab tricks for code speedup.
- Mex files for some Kronecker product computations.

Algorithm	Successes on Din (Out of 20)		Runtime on Din (ms / iter)	
	Original	Modified	Original	Modified
CH_LM_S	4	4	380	140
CH_LM_M	1	6	369	143
CH_LM_M_GN	15	19	205	109

C. Removal of redundant computations

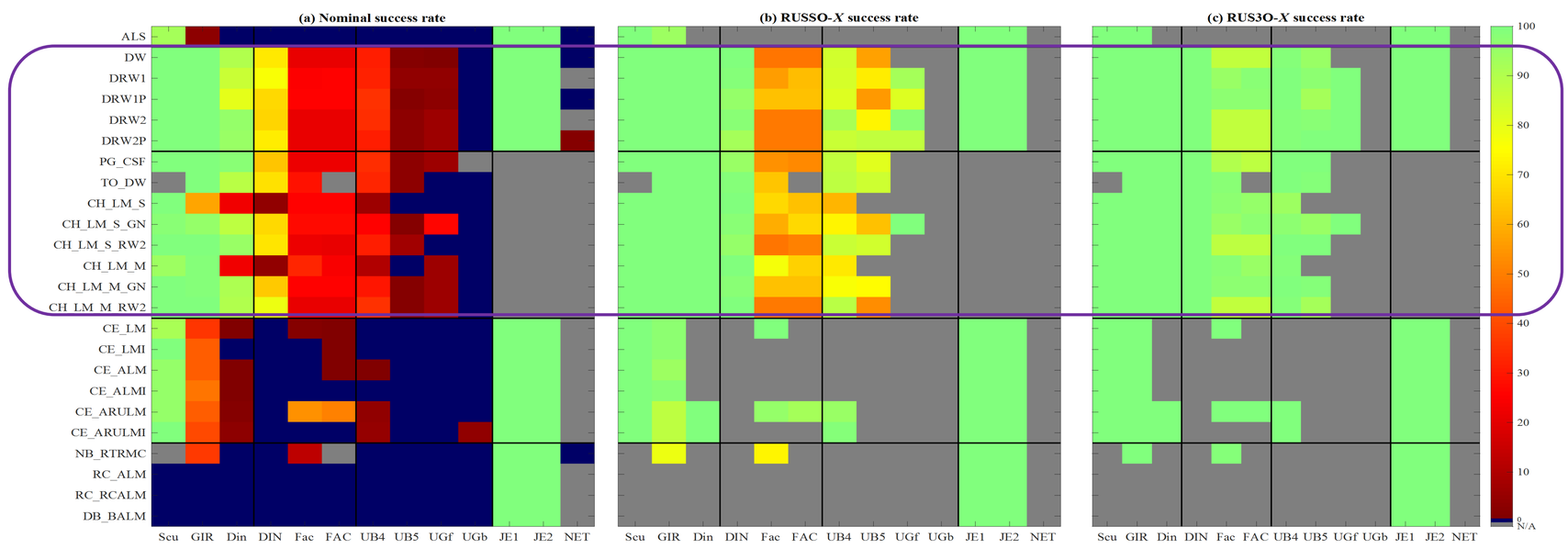
1. VarPro JTJ is of the form A $\otimes$ B where A and B are symmetric. i.e. internal symmetry in addition to the ordinary JTJ symmetry. (max. 4 fold speedup.)
2. $\widetilde{\mathrm{U}}_\mathrm{Q}$ is block-diagonal. Each sub-block $\widetilde{\mathrm{U}}_{\mathrm{j}\mathrm{Q}}$ is a function of the respective column of W; $\widetilde{\mathrm{U}}_{jQ} = qf\big(\widetilde{\mathrm{W}}_j\mathrm{U}\big)$ where $\widetilde{\mathrm{W}}_j = \mathrm{diag}(\mathrm{vec}\big(\mathrm{W}(:,j)\big)$. Hence, if $\mathrm{W}(:,i) = \mathrm{W}(:,j)$, $\widetilde{\mathrm{U}}_{iQ} = \widetilde{\mathrm{U}}_{jQ}$. i.e. redundant QR computations can be reduced depending on the dataset. (Up to 5 times faster for DIN)

8. A list of datasets used

ID	Nature	m	n	r	Fill (%)
Scu	Photometric stereo	46	16,301	3	66.5
GIR	Non-rigid SfM	166	240	6	69.8
Din	Rigid SfM	72	319	4	23.1
DIN	Rigid SfM	72	4,983	4	9.2
Fac	Photometric stereo	20	2,596	4	64.9
FAC	Photometric stereo	20	2,944	4	58.3
UB4	Non-rigid SfM	110	1,760	4	14.4
UB5	Non-rigid SfM	110	1,760	5	14.4
UGf	Non-rigid SfM	380	4,885	6	9.1
UGb	Non-rigid SfM	380	6,310	4	6.9
JE1	Recommender	100	24,983	7	72.5
JE2	Recommender	100	23,500	7	72.7
NET	Recommender	2,000	50,000	4	2.7

9. Comparison of success rates

When run from 20 to 100 different random starting points, the VarPro-based algorithms (row 2-14) perform best in terms of success rates. RUSSO-X and RUS3O-X improves it further. (RUSkO-X stands for "run algorithm-X until seen k^{th} optimum".)

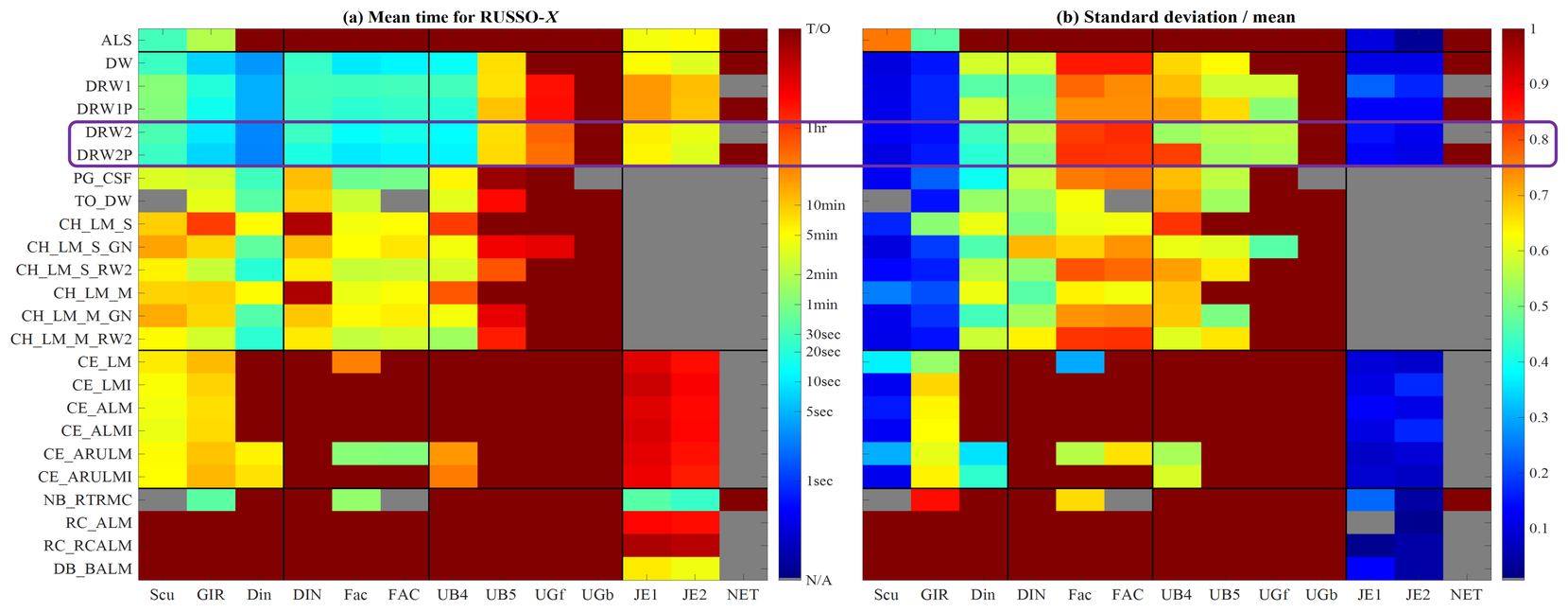


Conclusions

- **Approximations**: Recent top matrix factorization algorithms, which have been derived in a multitude of different ways, can be re-derived in a unified way such that similarities and differences are more easily observed.
- **Numerics**: Exploiting matrix structures and using block-QR decomposition improves the variable projection (VarPro) algorithms both in terms of speed and convergence.
- **Manifold optimization**: The Riemannian manifold optimization framework can be incorporated to give a slight more improvement.
- **Random restarts**: A simple meta-algorithm, RUSSO-X can be wrapped around any existing matrix factorization algorithm and improve its success rate.
- **Most importantly, VarPro is key to success** for matrix factorization, everything else is detail.

10. Mean time and standard deviation for RUSSO-X

All are run in single-threaded mode. Overall, the best performing family is the "DRW" set. DRW2 and DRW2P are the winners.



References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. Optimization Algorithms on Matrix Manifolds. Princeton University Press, 2008.
[2] N. Boumal and P.-A. Absil. RTRMC: A Riemannian trust-region method for low-rank matrix completion. NIPS 2011.
[3] A. M. Buchanan and A. W. Fitzgibbon. Damped Newton Algorithms for Matrix Factorization with Missing Data. CVPR, 2005.
[4] R. Cabral, F. De la Torre, J. P. Costeira, and A. Bernardino. Unifying Nuclear Norm and Bilinear Factorization Approaches for Low-rank Matrix Decomposition. ICCV, 2013.
[5] P. Chen. Optimization Algorithms on Subspaces: Revisiting Missing Data Problem in Low-rank Matrix. IJCV, 2008.
[6] G. H. Golub and V. Pereyra. The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate. SINUM, 1973.
[7] P. F. Gotardo and A. M. Martinez. Computing Smooth Time Trajectories for Camera and Deformable Shape in Structure from Motion with Occlusion. PAMI, 2011.
[8] R. Vidal and R. Hartley. Motion Segmentation with Missing Data Using PowerFactorization and GPCA.
[9] T. Okatani, T. Yoshida, and K. Deguchi. Efficient Algorithm for Low-rank Matrix Factorization with Missing Components and Performance Comparison of Latest Algorithms.. ICCV, 2011.
[10] A. Ruhe and P.A. Wedin. Algorithms for Separable Nonlinear Least Squares Problems. SIREV, 1980.
[11] T. Wiberg. Computation of principal components when data are missing. 1976.