



1. Separable Nonlinear Least Squares (SNLS) problems

A general SNLS problem has the form

$$\min_{\mathbf{u}, \mathbf{v}} \|\boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v})\|_2^2 = \min_{\mathbf{u}, \mathbf{v}} \|\mathbf{G}(\mathbf{u})\mathbf{v} - \mathbf{z}(\mathbf{u})\|_2^2 \quad (1)$$

where $\mathbf{u} \in \mathbb{R}^p$ and $\mathbf{v} \in \mathbb{R}^q$ are the model parameters and
 $\mathbf{z}: \mathbb{R}^q \rightarrow \mathbb{R}^s$ and $\mathbf{G}: \mathbb{R}^p \rightarrow \mathbb{R}^{s \times q}$ are functions of \mathbf{u}

Examples: Bundle adjustment (BA) with weak-perspective cameras [1]

$$\min_{\{\mathbf{q}_i\}, \{\mathbf{x}_j\}} \sum_{(i,j) \in \Omega} \|\mathbf{K}\mathbf{P}(\mathbf{q}_i)\tilde{\mathbf{x}}_j - \mathbf{m}_{ij}\|_2^2$$

BA with affine cameras [2]:

$$\min_{\{\mathbf{P}_i\}, \{\mathbf{x}_j\}} \sum_{(i,j) \in \Omega} \|\mathbf{P}_i\tilde{\mathbf{x}}_j - \mathbf{m}_{ij}\|_2^2$$

Other problems which can be cast as matrix factorization [3]

e.g. non-rigid SfM, recommender systems: $\min_{\mathbf{U}, \mathbf{V}} \|\mathbf{W} \odot (\mathbf{UV} - \mathbf{M})\|_F^2$

2. Known approaches

A. Joint optimization [4]

- Minimize (1) over \mathbf{u} and \mathbf{v} simultaneously using a 2nd order optimizer, i.e. form a stacked vector $\mathbf{x} = [\mathbf{u}; \mathbf{v}]$ and solve $\min_{\mathbf{x}=[\mathbf{u}; \mathbf{v}]} \|\boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v})\|_2^2$
- Embedded Point Iterations (EPI) [5,6]: after each joint update, one may perform further update on \mathbf{v} , i.e. $\mathbf{v}^*(\mathbf{u} + \Delta\mathbf{u}) = \operatorname{argmin}_{\mathbf{v}} \|\mathbf{G}(\mathbf{u} + \Delta\mathbf{u})\mathbf{v} - \mathbf{z}(\mathbf{u} + \Delta\mathbf{u})\|_2^2$

B. Variable Projection (VarPro) [7]

- Eliminate \mathbf{v} optimally from (1) and solve the reduced problem using a 2nd order optimizer, i.e. $\mathbf{v}^*(\mathbf{u}) := \operatorname{argmin}_{\mathbf{v}} \|\mathbf{G}(\mathbf{u})\mathbf{v} - \mathbf{z}(\mathbf{u})\|_2^2 = \mathbf{G}(\mathbf{u})^\dagger \mathbf{z}(\mathbf{u})$. Now solve

$$\min_{\mathbf{u}} \|\boldsymbol{\varepsilon}^*(\mathbf{u})\|_2^2 := \min_{\mathbf{u}} \|\boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v}^*(\mathbf{u}))\|_2^2 = \min_{\mathbf{u}} \|\mathbf{G}(\mathbf{u}) \mathbf{G}(\mathbf{u})^\dagger \mathbf{z}(\mathbf{u}) - \mathbf{z}(\mathbf{u})\|_2^2$$

3. Levenberg-Marquardt (LM) algorithm [8]

LM: 2nd-order optimization algorithm for solving general nonlinear least squares problem

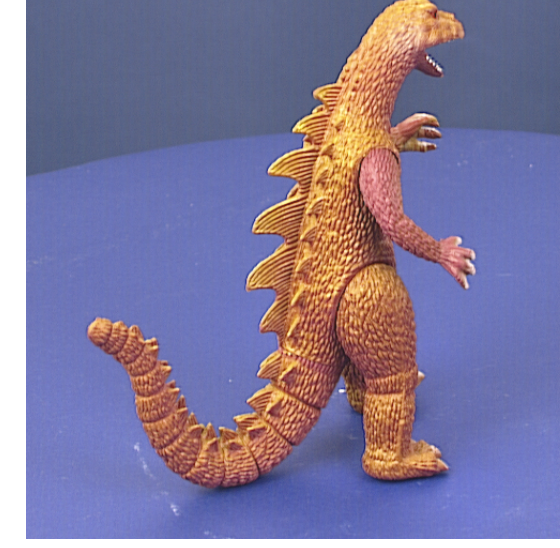
$$\min_{\mathbf{x}} \|\boldsymbol{\varepsilon}(\mathbf{x})\|_2^2$$

- It assumes the residual vector $\boldsymbol{\varepsilon}(\mathbf{x})$ to be locally linear. i.e. $\boldsymbol{\varepsilon}(\mathbf{x} + \Delta\mathbf{x}) \approx \boldsymbol{\varepsilon}(\mathbf{x}) + \frac{\partial \boldsymbol{\varepsilon}(\mathbf{x})}{\partial \mathbf{x}} \Delta\mathbf{x}$
- Damping (λ) penalizes large updates
It implicitly decides the trust region radius

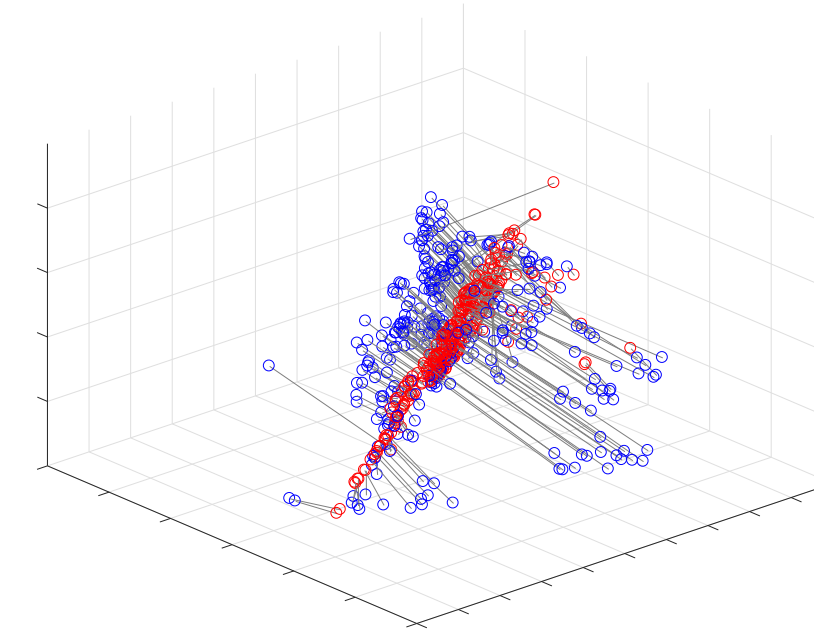
```
1: inputs:  $\mathbf{x} \in \mathbb{R}^n$ 
2:  $\lambda \leftarrow 10^{-4}$ 
3: repeat
4:   repeat
5:      $\Delta\mathbf{x} \leftarrow \operatorname{argmin}_{\Delta\mathbf{x}} \left\| \boldsymbol{\varepsilon}(\mathbf{x}) + \frac{\partial \boldsymbol{\varepsilon}(\mathbf{x})}{\partial \mathbf{x}} \Delta\mathbf{x} \right\|_2^2 + \lambda \|\Delta\mathbf{x}\|_2^2$ 
6:      $\lambda \leftarrow 10\lambda$ 
7:   until  $\|\boldsymbol{\varepsilon}(\mathbf{x} + \Delta\mathbf{x})\|_2^2 < \|\boldsymbol{\varepsilon}(\mathbf{x})\|_2^2$ 
8:    $\mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}$ 
9:    $\lambda \leftarrow 0.01\lambda$ 
10: until convergence
11: output:  $\mathbf{x} \in \mathbb{R}^n$ 
```

4. VarPro vs Joint optimization on SNLS problems

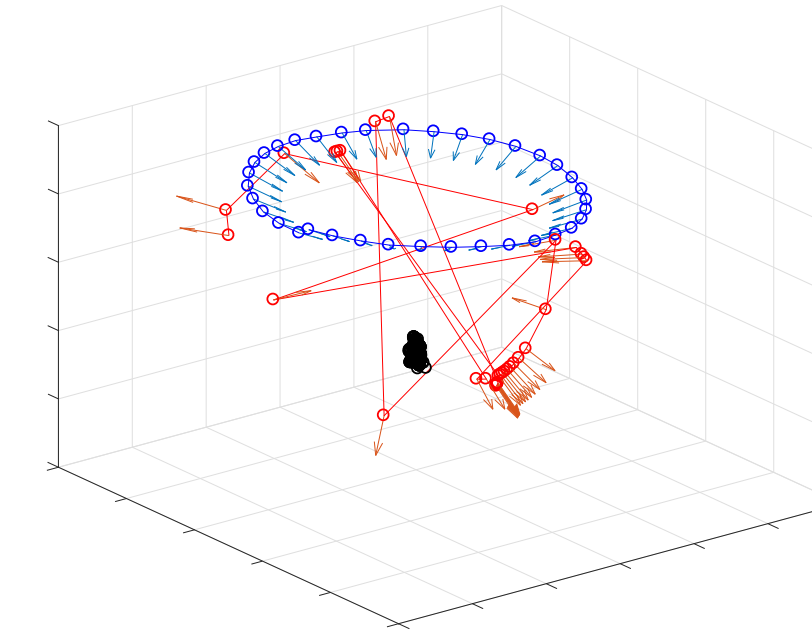
It is widely known that VarPro (blue) outperforms Joint optimization (red) [3,9,10]



(a) Trimmed dinosaur [3,4]



(b) 3D structure



(c) Affine cameras

A. Common misconceptions

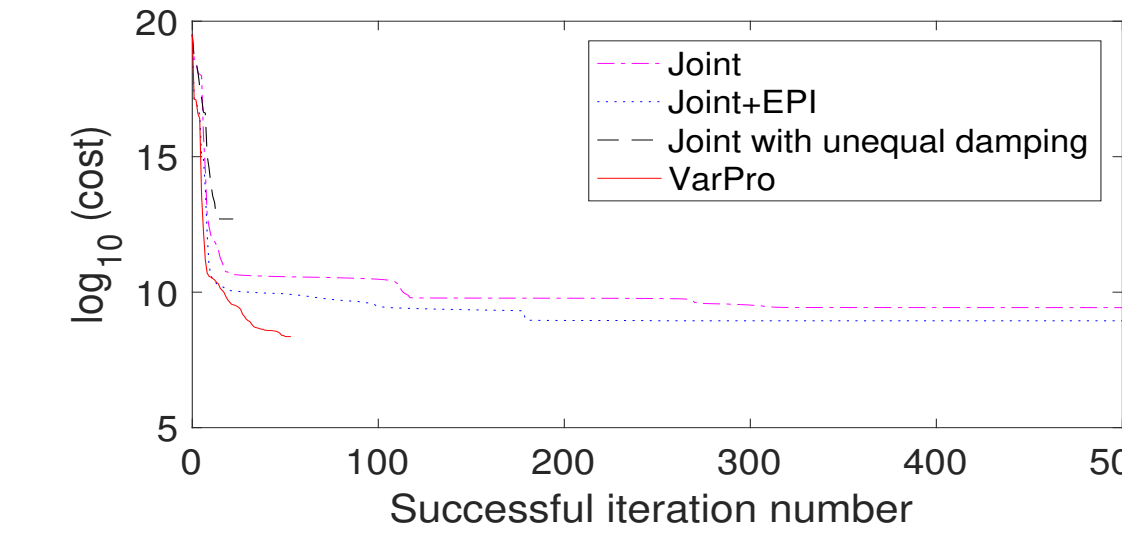
- VarPro has higher iteration complexity than Joint optimization (Joint) and is not scalable
- VarPro = Joint Optimization plus EPI (Joint+EPI) [6]

B. Our comparison strategy and results

- Analysis of the update equations of \mathbf{u} and \mathbf{v} (for the LM subproblem) for each method
- More details in the paper

	$\lambda_{\mathbf{v}} \neq 0$	$\lambda_{\mathbf{v}} = 0$
EPI off	Joint (4%)	(Joint+zero $\lambda_{\mathbf{v}}$) (0%)
EPI on	Joint+EPI (24%)	VarPro (94 %)

(d) Comparison of methods on (a)



(e) Iteration plots of methods in (d)

D. Unified pseudocode

Joint	Joint+EPI	VarPro	inputs: $\mathbf{u} \in \mathbb{R}^p, \mathbf{v} = \operatorname{argmin}_{\mathbf{v}} \ \boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v})\ _2^2 \in \mathbb{R}^q$
•	•	•	1: $[\lambda_{\mathbf{u}}; \lambda_{\mathbf{v}}] \leftarrow [10^{-4}; 10^{-4}]$
•	•	•	2: $\lambda_{\mathbf{v}} \leftarrow 0$
•	•	•	3: repeat
•	•	•	4: repeat
•	•	•	5: $\mathbf{g} \leftarrow \mathbf{J}_{\mathbf{u}}(\mathbf{u}, \mathbf{v})^\top (\mathbf{I} - \mathbf{J}_{\mathbf{v}}(\mathbf{u}) \mathbf{J}_{\mathbf{v}}(\mathbf{u})^{-\lambda_{\mathbf{v}}}) \boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v})$
•	•	•	6: $\Delta\mathbf{u} \leftarrow -(\mathbf{J}_{\mathbf{u}}(\mathbf{u}, \mathbf{v})^\top (\mathbf{I} - \mathbf{J}_{\mathbf{v}}(\mathbf{u}) \mathbf{J}_{\mathbf{v}}(\mathbf{u})^{-\lambda_{\mathbf{v}}}) \mathbf{J}_{\mathbf{u}}(\mathbf{u}, \mathbf{v}) + \lambda_{\mathbf{u}} \mathbf{I})^{-1} \mathbf{g}$
•	•	•	7: $\Delta\mathbf{v} \leftarrow \operatorname{argmin}_{\Delta\mathbf{v}} \ \boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v}) + \mathbf{J}_{\mathbf{u}}(\mathbf{u}, \mathbf{v}) \Delta\mathbf{u} + \mathbf{J}_{\mathbf{v}}(\mathbf{u}) \Delta\mathbf{v}\ _2^2 + \lambda_{\mathbf{v}} \ \Delta\mathbf{v}\ _2^2$
•	•	•	8: $\Delta\mathbf{v} \leftarrow \operatorname{argmin}_{\Delta\mathbf{v}} \ \boldsymbol{\varepsilon}(\mathbf{u} + \Delta\mathbf{u}, \mathbf{v}) + \mathbf{J}_{\mathbf{v}}(\mathbf{u} + \Delta\mathbf{u}) \Delta\mathbf{v}\ _2^2$
•	•	•	9: $[\lambda_{\mathbf{u}}; \lambda_{\mathbf{v}}] \leftarrow 10 [\lambda_{\mathbf{u}}; \lambda_{\mathbf{v}}]$
•	•	•	10: until $\ \boldsymbol{\varepsilon}(\mathbf{u} + \Delta\mathbf{u}, \mathbf{v} + \Delta\mathbf{v})\ _2^2 < \ \boldsymbol{\varepsilon}(\mathbf{u}, \mathbf{v})\ _2^2$
•	•	•	11: $[\mathbf{u}; \mathbf{v}] \leftarrow [\mathbf{u}; \mathbf{v}] + [\Delta\mathbf{u}; \Delta\mathbf{v}]$
•	•	•	12: $[\lambda_{\mathbf{u}}; \lambda_{\mathbf{v}}] \leftarrow 0.01 [\lambda_{\mathbf{u}}; \lambda_{\mathbf{v}}]$
•	•	•	13: until convergence
			output: $\mathbf{u} \in \mathbb{R}^p, \mathbf{v} \in \mathbb{R}^q$

E. Scalable implementation using MINRES

- Iteration complexity of VarPro is roughly equal to that of Joint+EPI
- Any indirect solvers such as Preconditioned Conjugate Gradient (PCG) may be used to solve the LM subproblem of VarPro
- MINRES (similar to PCG) shows better numerical stability and is thus selected

5. Experimental results

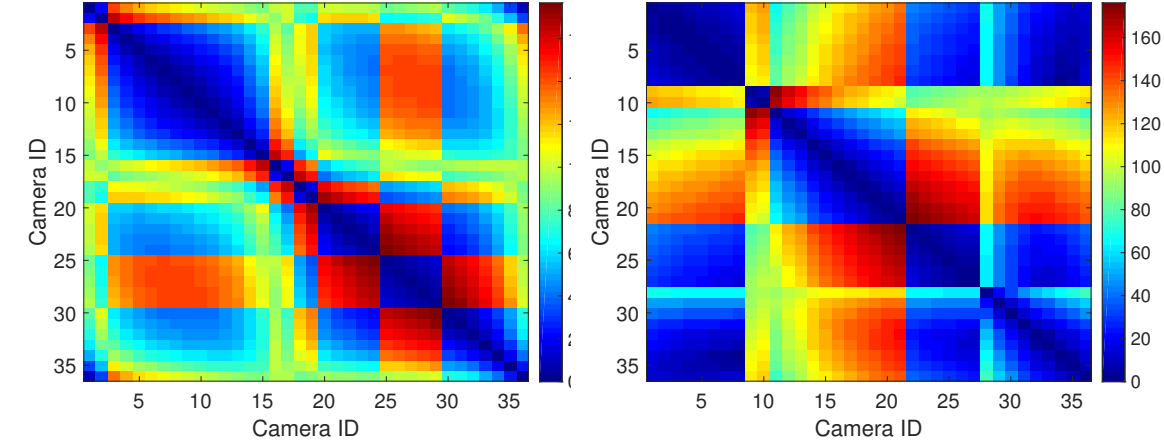
Dataset	f	n	Missing (%)	Joint	Joint+EPI	VarPro	VarPro-MinRes*
Blue teddy bear (trimmed)	196	827	80.7	10 (238)	20 (155)	88 (22.3)	76 (21.9)
Corridor	11	737	50.2	40 (8.71)	4 (14.8)	100 (1.07)	100 (0.78)
Dinosaur (trimmed)	36	319	76.9	4 (5.95)	24 (9.38)	94 (1.55)	99 (3.96)
Dinosaur including outliers	36	4983	90.8	0 (28.6)	0 (62.1)	100 (13.9)	36 (38.9)
House	10	672	57.7	44 (4.90)	8 (9.71)	100 (0.30)	100 (0.41)
Road scene #47	11	150	47.1	44 (1.88)	32 (3.00)	100 (0.16)	100 (0.17)
Stockholm Guildhall (trimmed)	43	1000	18.0	92 (45.1)	48 (35.7)	100 (22.8)	100 (3.12)
Wilshire	190	411	60.7	38 (409)	94 (9.90)	100 (7.64)	100 (1.96)
Ladybug (skeleton)	49	7776	91.6	0 (77.3)	0 (155)	50 (49.7)	0 (155)
Trafalgar Square (skeleton)	21	11315	84.7	0 (76.2)	0 (160)	100 (14.7)	100 (56.4)
Dubrovnik (skeleton)	16	22106	76.3	38 (159)	0 (346)	100 (23.6)	100 (32.9)
Venice (skeleton)	52	64053	89.6	0 (913)	0 (1495)	80 (123)	60 (329)

Table 1: Success rates of different methods with median times in parenthesis

- VarPro-MINRES implemented less efficiently in MATLAB
- VarPro implemented using a patched version of Ceres Solver [6]

6. Why does Joint opt. perform poorly on SNLS problems?

- Joint+EPI and VarPro behave similarly unless $\mathbf{v}^*(\mathbf{u})$ (triangulated 3D points) is sensitive to changes in \mathbf{u} (cameras)
- $\mathbf{G}(\mathbf{u})$ is poorly conditioned in this case
Almost parallel cameras in affine BA
- Damping on the eliminated parameters \mathbf{v} ($\lambda_{\mathbf{v}}$) discourages large updates on \mathbf{v} (3D points)
- On the other hand, VarPro allows large updates of \mathbf{v} , therefore improving the update $\Delta\mathbf{u}$



(a) VarPro (94%) (b) Joint+EPI

Angles between pairs of affine camera directions in (c)

7. Conclusion

- Joint optimization and VarPro are algorithmically very similar and can be unified
- The most important difference between Joint optimization and VarPro is the unbalanced trust-region assumption in VarPro
- VarPro can be in principle implemented as efficiently as standard Joint optimization
- VarPro has much higher success rates than Joint optimization in affine BA due to its tolerance to large updates in 3D points when cameras are near parallel

Acknowledgement

- Microsoft, Toshiba Research Europe and Roberto Cipolla for funding support.
- Microsoft, Christ's College and Cambridge Philosophical Society for travel subsistence.

References

- | | |
|--|---|
| [1] Y. Zheng et al., CVPR 2012 | [6] S. Agarwal, K. Mierle and Others, 2017 |
| [2] C. Tomasi and T. Kanade, IJCV 1992 | [7] G. Golub and V. Pereyra, SINUM 1973 |
| [3] J.H. Hong and A.W. Fitzgibbon, ICCV 2015 | [8] K. Levenberg, Quart. Appl. Math., 1944 |
| [4] A. Buchanan and A.W. Fitzgibbon, CVPR 2005 | [9] T. Okatani et al., ICCV 2011 |
| [5] Y. Jeong et al., TPAMI 2012 | [10] D.P. O'Leary and B. Rust, Comp. Opt. Appl., 2013 |