Algorithms for 3D-3D Registration with Known and Unknown References: Applications to Materials Science

$$\min \left\| \overset{\text{reference configuration}}{\underset{N_{ref}\,\mathrm{x}d}{\boxed{\underline{\underline{X}}}}} - \overset{\text{permutation matrix}}{\underset{N_{ref}\,\mathrm{x}N_{nbr}}{\boxed{\underline{\underline{P}}_i}}} \overset{\text{sample configs.}}{\underset{N_{nbr}\mathrm{x}d}{\boxed{\underline{\underline{Y}}_i}}} \overset{\text{rotation matrix}}{\underset{d\mathrm{x}d}{\boxed{\underline{\underline{R}}_i}}} \right\|_F$$

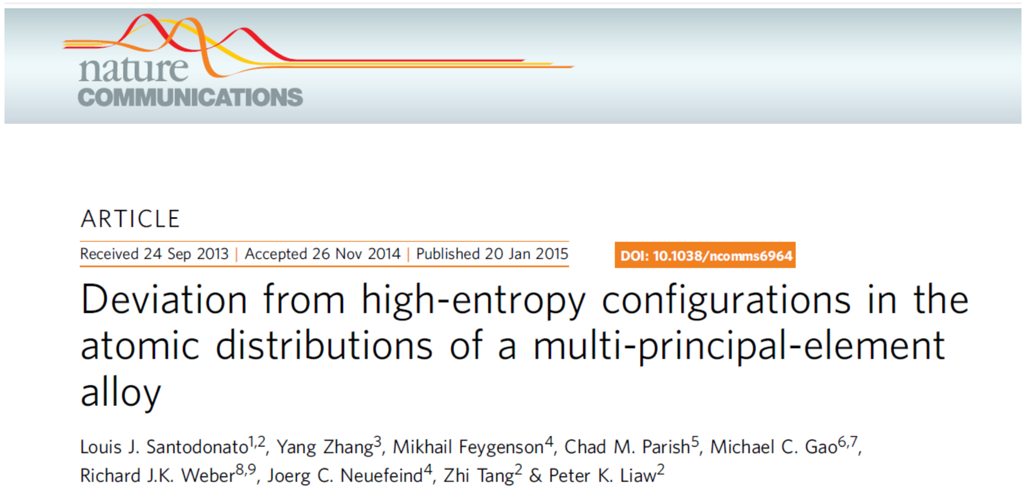
David Keffer & Nick McNutt

University of Tennessee

Presented to the Innovative Computing Laboratory

January 15, 2016

Motivation from Atomic Probe Tomography


nature COMMUNICATIONS

ARTICLE

Received 24 Sep 2013 | Accepted 26 Nov 2014 | Published 20 Jan 2015 DOI: 10.1038/ncomms6964

Deviation from high-entropy configurations in the atomic distributions of a multi-principal-element alloy

Louis J. Santodonato[1,2], Yang Zhang[3], Mikhail Feygenson[4], Chad M. Parish[5], Michael C. Gao[6,7], Richard J.K. Weber[8,9], Joerg C. Neuefeind[4], Zhi Tang[2] & Peter K. Liaw[2]

data set of 10^7 atoms
atomic identity & coordinates

Material under study
High Entropy Alloy (HEA)
$Al_{1.3}CoCrCuFeNi$

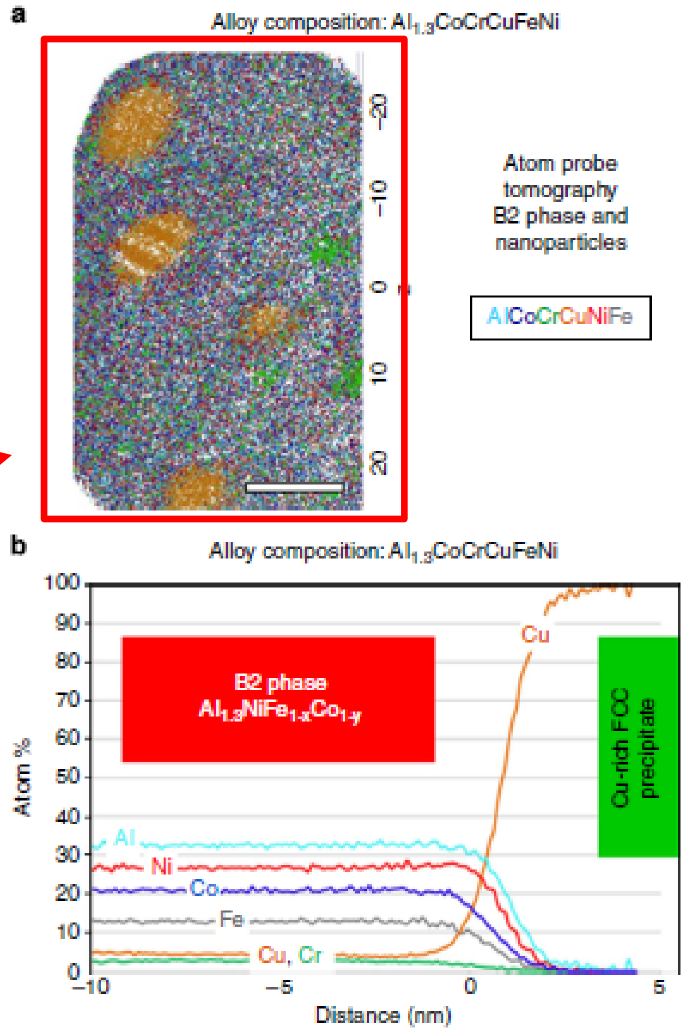
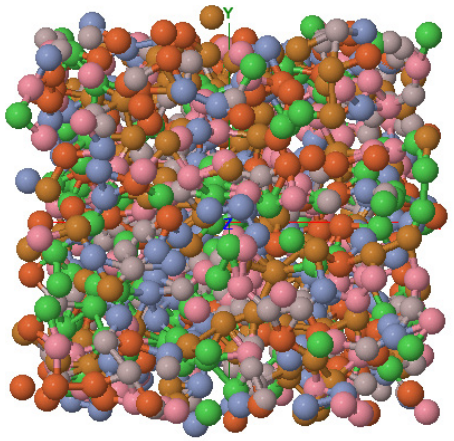
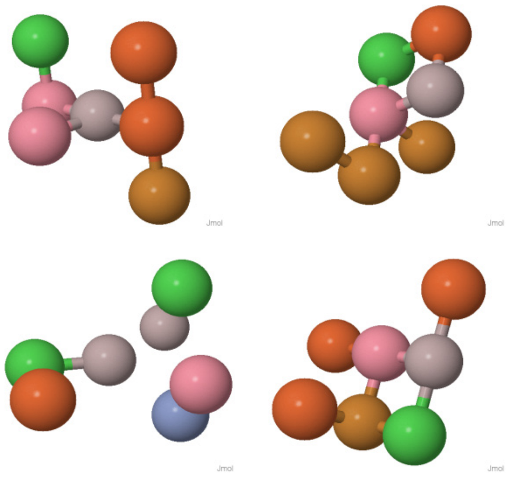


Figure 3 | APT. (**a**) An atom map shows the atomic distribution within the B2 phase, which contains Cu-rich and Cr-rich nanoprecipitates. Scale bar, 0.01 μm. (**b**) Proximity histogram presenting atomic concentrations across the B2–FCC interface.

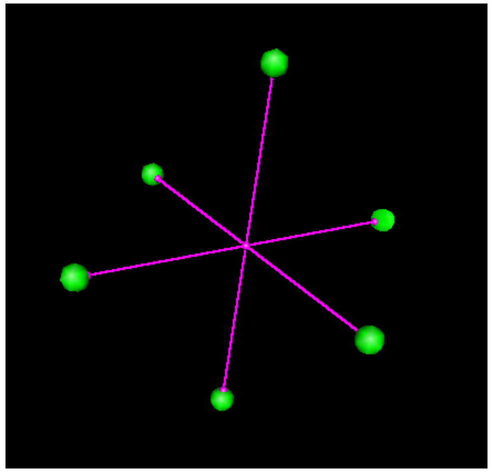
Overview of Process



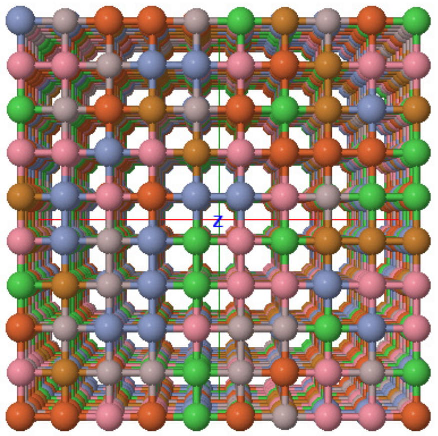
data set: missing 2/3 atoms
and subject to xyz uncertainty



local neighbors
(four shown w/ N_{nbr} = 6)



reconstructed local neighborhood
average and distribution



true structure:
spatially & chemical ordering

Problem Statement: Minimize the Frobenius norm of the following matrix:

$$\min \left\| \overset{\text{reference configuration}}{\underset{N_{ref}\,\mathrm{x}d}{\boxed{\underline{\underline{X}}}}} - \overset{\text{permutation matrix}}{\underset{N_{ref}\,\mathrm{x}N_{nbr}}{\boxed{\underline{\underline{P}}_i}}} \overset{\text{sample configs.}}{\underset{N_{nbr}\,\mathrm{x}d}{\boxed{\underline{\underline{Y}}_i}}} \overset{\text{rotation matrix}}{\underset{d\mathrm{x}d}{\boxed{\underline{\underline{R}}_i}}} \right\|_F$$

d = dimensionality, e.g. d = 3
N_{nbr} = number of points in each configuration, e.g. $N_{nbr} \sim 12$
N_{ref} = number of points in reference configuration, e.g. $N_{ref} \sim 50$
m = number of configurations, e.g. m = 10^7 per frame, ??? frames

Permutation matrix: Review

Permutation matrices change the order of rows.
They are filled with ones and zeros, e.g.

$$P_\pi = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \mathbf{e}_{\pi(3)} \\ \mathbf{e}_{\pi(4)} \\ \mathbf{e}_{\pi(5)} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_4 \\ \mathbf{e}_2 \\ \mathbf{e}_5 \\ \mathbf{e}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Given a vector **g**,

$$P_\pi \mathbf{g} = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \mathbf{e}_{\pi(3)} \\ \mathbf{e}_{\pi(4)} \\ \mathbf{e}_{\pi(5)} \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_4 \\ g_2 \\ g_5 \\ g_3 \end{bmatrix}.$$

Permutation matrices don’t have to be square.
You can put n rows in m>n slots.

https://en.wikipedia.org/wiki/Permutation_matrix

Rotation Matrix: Review

Rotation matrices rotate points in d-dimensional space.
They are orthonormal matrices.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The above matrices rotate a point about the x, y and z axes respectively an angle of θ radians.

https://en.wikipedia.org/wiki/Rotation_matrix

Two Versions of the Problem

Version 1. “Easier Problem”, defined by the fact that $\underline{\underline{X}}$ is known.

$$\min \left\| \overset{\text{known}}{\underline{\underline{X}}} - \overset{\text{unknown}}{\underline{\underline{P}}_i}\, \overset{\text{known}}{\underline{\underline{Y}}_i}\, \overset{\text{unknown}}{\underline{\underline{R}}_i} \right\|_F$$

N_{ref} x d $\quad$ N_{ref} x N_{nbr} $\quad$ N_{nbr} x d $\quad$ d x d

This problem can be solved independently for each of the i = 1 to m ~ 10^7

Embarrassingly parallel implementation

What is algorithm?
First consider two even simpler problems….

Version 1.a. “Easiest Problem”, defined by the fact that $\underline{\underline{X}}$ and $\underline{\underline{R}}_i$ are known.

$$\min \left\| \underset{N_{ref}\,\mathrm{x}d}{\overset{\text{known}}{\underline{\underline{X}}}} - \underset{N_{ref}\,\mathrm{x}N_{nbr}}{\overset{\color{red}{\text{unknown}}}{\underline{\underline{P}}_i}} \; \underset{N_{nbr}\,\mathrm{x}d}{\overset{\text{known}}{\underline{\underline{Y}}_i}} \; \underset{d\mathrm{x}d}{\overset{\text{known}}{\underline{\underline{R}}_i}} \right\|_F$$

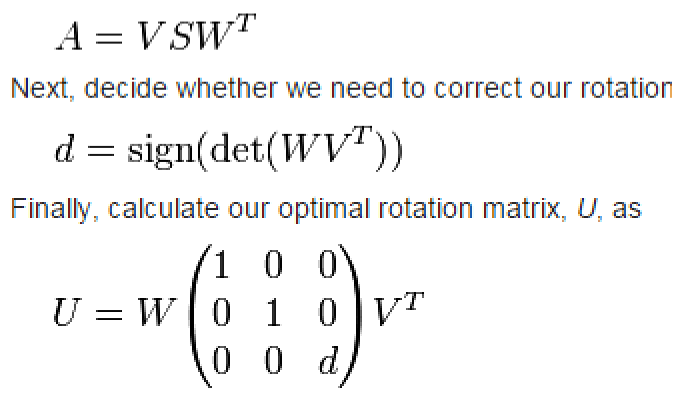
If the only unknown is the permutation matrix, this problem is known as “the non-square assignment problem” and is solved via the “Hungarian Algorithm” or sometimes as the “Kuhn-Munkres” algorithm. This is a non-iterative linear algebra algorithm.

Kuhn, H.W., *The Hungarian Method for the assignment problem.* Naval Research Logistics Quarterly, 1955. **2**: p. 83–97.
https://en.wikipedia.org/wiki/Hungarian_algorithm.

Version 1.b. “Easiest Problem”, defined by the fact that $\underline{\underline{X}}$ and $\underline{\underline{P}}_i$ are known.

$$\min \left\| \underset{N_{ref}\,\mathrm{x}d}{\overset{\text{known}}{\underline{\underline{X}}}} - \underset{N_{ref}\,\mathrm{x}N_{nbr}}{\overset{\text{known}}{\underline{\underline{P}}_i}} \; \underset{N_{nbr}\,\mathrm{x}d}{\overset{\text{known}}{\underline{\underline{Y}}_i}} \; \underset{d\mathrm{x}d}{\overset{\text{unknown}}{\underline{\underline{R}}_i}} \right\|_F$$

If the only unknown is the rotation matrix, this problem is simply singular value decomposition SVD. When SVD generates a rotation matrix, this is called the Kabsch algorithm. This is a non-iterative linear algebra algorithm.



$$A = VSW^T$$

Next, decide whether we need to correct our rotation

$$d = \operatorname{sign}(\det(WV^T))$$

Finally, calculate our optimal rotation matrix, U, as

$$U = W \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & d \end{pmatrix} V^T$$

https://en.wikipedia.org/wiki/Singular_value_decomposition
https://en.wikipedia.org/wiki/Kabsch_algorithm

Version 1. "Easier Problem", defined by the fact that $\underline{\underline{X}}$ is known.

$$\min \left\| \overset{\text{known}}{\underline{\underline{X}}} - \overset{\color{red}{\text{unknown}}}{\underline{\underline{P}}_i} \overset{\text{known}}{\underline{\underline{Y}}_i} \overset{\color{red}{\text{unknown}}}{\underline{\underline{R}}_i} \right\|_F$$

N_{ref} xd $\quad N_{ref}$ xN_{nbr} $\quad N_{nbr}$ xd $\quad$ dxd

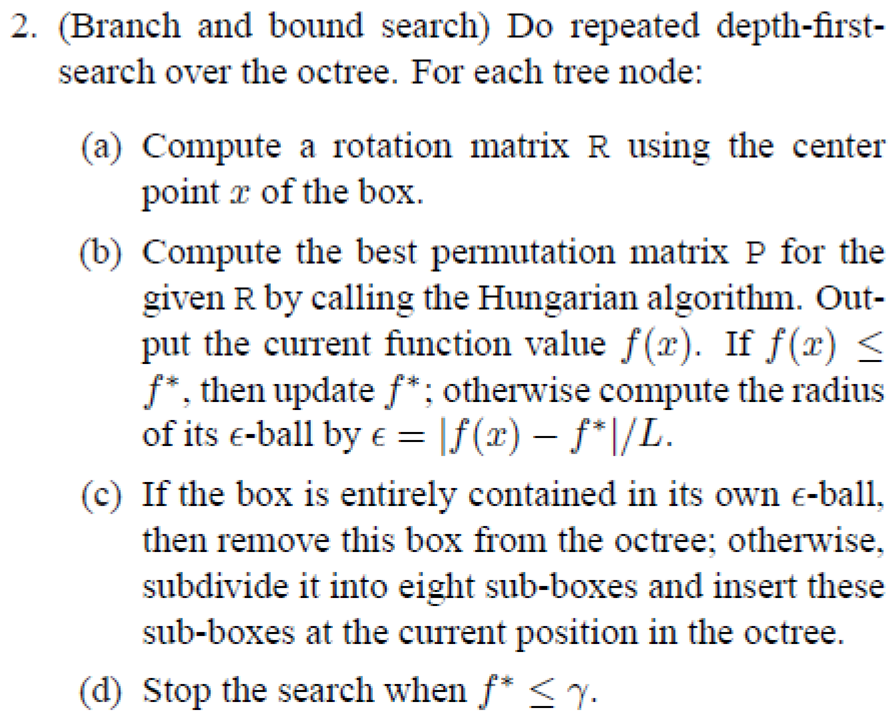
When both permutation and rotation matrices are unknown, this is called the "3D-3D registration" problem.

Our approach to implementing this solution is taken from the following work:

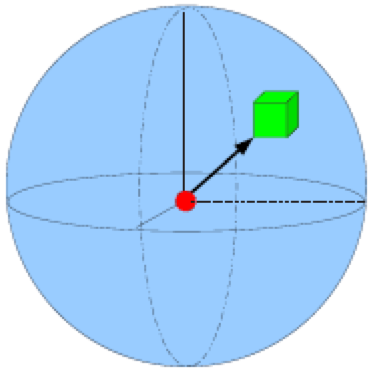
Li, H.D. and R. Hartley. *The 3D-3D Registration Problem Revisited* in *ICCV 2007: Eleventh IEEE International Conference on Computer Vision*, 2007, Rio de Janeiro.

Version 1. “Easier Problem”, defined by the fact that $\underline{\underline{X}}$ is known.

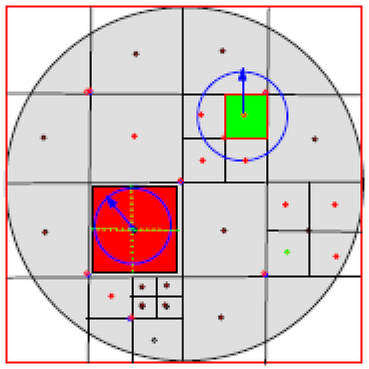
Approach: Search 3D rotation space. At each point evaluated, use Hungarian algorithm to compute best permutation matrix. Find global minimum in Frobenius norm by an Octree-based search that uses a branch and bound algorithm.



2. (Branch and bound search) Do repeated depth-first-search over the octree. For each tree node:
   (a) Compute a rotation matrix R using the center point x of the box.
   (b) Compute the best permutation matrix P for the given R by calling the Hungarian algorithm. Output the current function value $f(x)$. If $f(x) \leq f^*$, then update f^*; otherwise compute the radius of its ϵ-ball by $\epsilon = |f(x) - f^*|/L$.
   (c) If the box is entirely contained in its own ϵ-ball, then remove this box from the octree; otherwise, subdivide it into eight sub-boxes and insert these sub-boxes at the current position in the octree.
   (d) Stop the search when $f^* \leq \gamma$.



Rotation matrix expressed as 3D sphere w/ radius pi.



Branch and bound algorithm uses a hierarchical approach to eliminate some rotation volume during optimization process.

Li, H.D. and R. Hartley. *The 3D-3D Registration Problem, 2007.*

Version 1. “Easier Problem”, defined by the fact that $\underline{\underline{X}}$ is known.

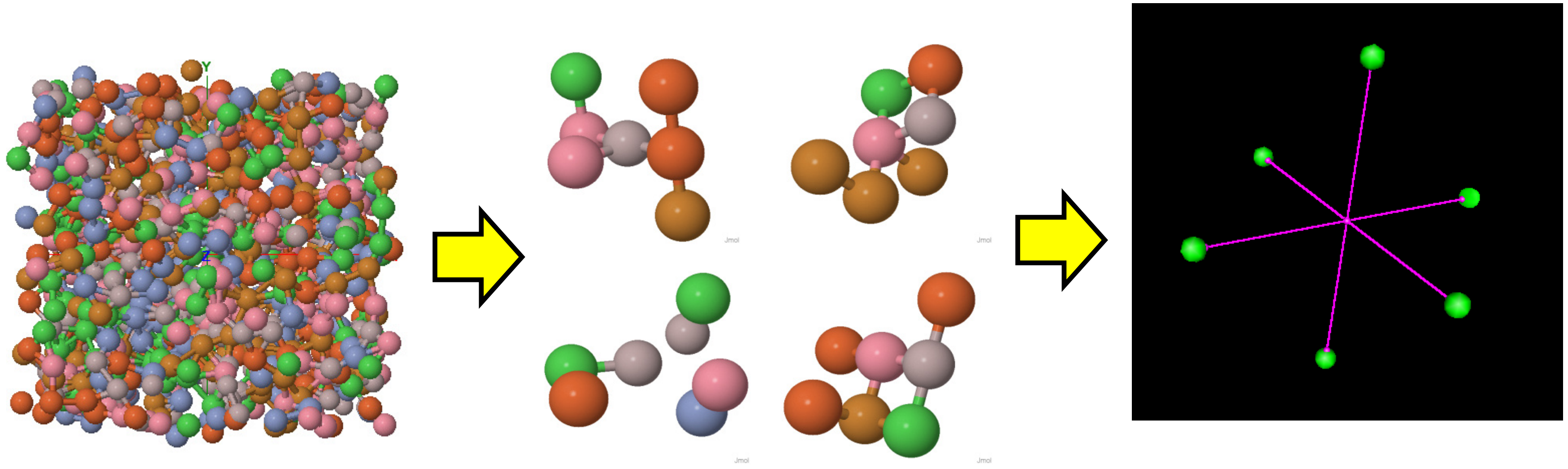
Current Status:

3D-3D registration problem implemented in serial.

Works well on artificial data sets w/ coordinate noise or missing atoms.

Although described as guaranteed to be globally convergent, sometimes finds local minima for real data sets.

Some work still to be done.
Bug in theory or implementation? Maybe fixed on 01/14/16…



Two Versions of the Problem

Version 2. “Harder Problem”, defined by the fact that $\underline{\underline{X}}$ is unknown.

$$\min \sum_{i=1}^{m} \left\| \underset{N_{ref}\,\mathrm{x}\,d}{\overset{\text{unknown}}{\underline{\underline{X}}}} - \underset{N_{ref}\,\mathrm{x}\,N_{nbr}}{\overset{\text{unknown}}{\underline{\underline{P}}_i}} \; \underset{N_{nbr}\,\mathrm{x}\,d}{\overset{\text{known}}{\underline{\underline{Y}}_i}} \; \underset{d\mathrm{x}d}{\overset{\text{unknown}}{\underline{\underline{R}}_i}} \right\|_F$$

This problem cannot be solved independently for each of the i = 1 to m ~ 10^7 configurations.
Requires parallel implementation

What is algorithm?

Reformulate minimization process as a trace maximization problem.

$$\arg\min_{\{R_i\},\{P_i\}} \sum_{i}^{m} \| \mathsf{X} - \mathsf{P}_i \mathsf{A}_i \mathsf{R}_i \|_F^2$$

$$\arg\max_{\{R_i\},\{P_i\}} \| \sum_{i}^{m} \mathsf{P}_i \mathsf{A}_i \mathsf{R}_i \|_F^2$$

$$\arg\max_{\{R_i\},\{P_i\}} \operatorname{tr} \sum_{i,j}^{m} \mathsf{A}_i^T (\mathsf{P}_i^T \mathsf{P}_j) \mathsf{A}_j (\mathsf{R}_j \mathsf{R}_i^T)$$

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Requires formulation of permutation and rotation "super matrices",

$$\underline{\underline{P}}_S = \begin{bmatrix} \underline{\underline{P}}_1 \underline{\underline{P}}_1^T & \underline{\underline{P}}_1 \underline{\underline{P}}_2^T & \cdots & \underline{\underline{P}}_1 \underline{\underline{P}}_m^T \\ \underline{\underline{P}}_2 \underline{\underline{P}}_1^T & \underline{\underline{P}}_2 \underline{\underline{P}}_2^T & & \\ \vdots & & \ddots & \\ \underline{\underline{P}}_m \underline{\underline{P}}_1^T & & & \underline{\underline{P}}_m \underline{\underline{P}}_m^T \end{bmatrix}$$

Each permutation matrix $\underline{\underline{P}}_i$ is N_{nbr} xd

Each $\underline{\underline{P}}_i \underline{\underline{P}}_j^T$ matrix is N_{nbr} x N_{nbr}

$1.2 \cdot 10^8$ x $1.2 \cdot 10^8$

The super permutation matrix $\underline{\underline{P}}_S$ is mN_{nbr} x mN_{nbr}

d = dimensionality, e.g. d = 3
N_{nbr} = number of points in each configuration, e.g. $N_{nbr} \sim 12$
N_{ref} = number of points in reference configuration, e.g. $N_{ref} \sim 50$
m = number of configurations, e.g. m = 10^7 per frame, ??? frames

Version 2. “Harder Problem”, defined by the fact that $\underline{\underline{X}}$ is unknown.

Requires formulation of permutation and rotation “super matrices”,

$$\underline{\underline{R}}_S = \begin{bmatrix} \underline{\underline{R}}_1 \underline{\underline{R}}_1^T & \underline{\underline{R}}_1 \underline{\underline{R}}_2^T & \cdots & \underline{\underline{R}}_1 \underline{\underline{R}}_m^T \\ \underline{\underline{R}}_2 \underline{\underline{R}}_1^T & \underline{\underline{R}}_2 \underline{\underline{R}}_2^T & & \\ \vdots & & \ddots & \\ \underline{\underline{R}}_m \underline{\underline{R}}_1^T & & & \underline{\underline{R}}_m \underline{\underline{R}}_m^T \end{bmatrix}$$

Each rotation matrix $\underline{\underline{R}}_i$ is d xd

Each $\underline{\underline{R}}_i \underline{\underline{R}}_j^T$ matrix is d x d

$3{\cdot}10^7$ x $3{\cdot}10^7$

The super rotation matrix $\underline{\underline{R}}_S$ is md x md

d = dimensionality, e.g. d = 3
N_{nbr} = number of points in each configuration, e.g. N_{nbr} ~ 12
N_{ref} = number of points in reference configuration, e.g. N_{ref} ~ 50
m = number of configurations, e.g. m = 10^7 per frame, ??? frames

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Step 1. Initially populate $\underline{\underline{P}}_S$ and $\underline{\underline{R}}_S$ as follows:

Solve "Easier problem" on a pairwise basis, $\underline{\underline{X}} = \underline{\underline{Y}}_j$

$$\min \left\| \underline{\underline{Y}}_j - \underline{\underline{P}}_{ij} \underline{\underline{Y}}_i \underline{\underline{R}}_{ij} \right\|_F$$

These pairwise rotation and permutation matrices can be used to initially populate the supermatrices.

They provide a good guess because the matrix contains all pairwise information.

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Step 2. Solve for converged $\underline{\underline{P}}_S$ and $\underline{\underline{R}}_S$ independently.

This is Nick's hypothesis.

It can be shown to work identically for randomly permuted and rotated data sets with no noise.

We have also some empirical evidence that an algorithm that relies on this hypothesis works well for systems with noise.

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Step 3. Solve for converged. $\underline{\underline{P}}_S$

3.1. Compute N_{nbr} leading eigenvectors of this matrix.
Begin loop over m configurations
    3.2. Use this to compute something close to permutation matrix, but not all zeros and ones.
    3.3. Then "clean this up" to become a proper permutation matrix, using Hungarian algorithm.
End loop over m configurations

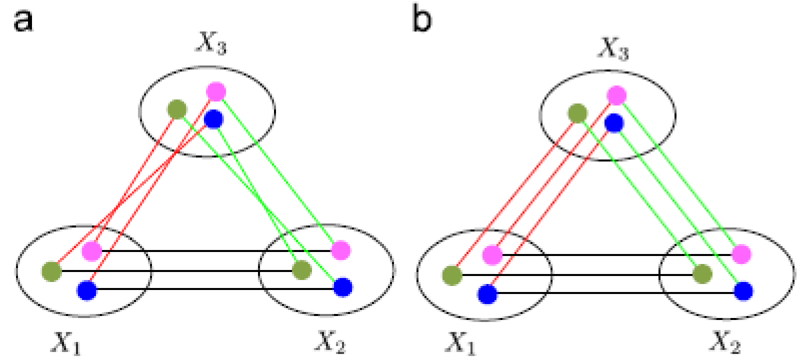


Fig. 1. Illustration of globally consistent correspondence (taking three feature sets X_1, X_2 and X_3 for an example). (a) Globally inconsistent correspondence; (b) globally consistent correspondence. Feature points in the same color correspond to the same entity.

Algorithm 1 Permutation Synchronization

```
Input: the objective matrix $\mathcal{T}$
  Compute the $n$ leading eigenvectors $(v_1, v_2, \ldots, v_n)$ of $\mathcal{T}$ and set $U = \sqrt{m}\,[v_1, v_2, \ldots, v_n]$
  for $i = 1$ to $m$ do
    $P_{i1} = U_{(i-1)n+1:in,\,1:n}\, U_{1:n,\,1:n}^{\top}$
    $\sigma_i = \arg\max_{\sigma \in \mathbb{S}_n} \langle P_{i1}, \sigma \rangle$  [Kuhn-Munkres]
  end for
  for each $(i, j)$ do
    $\tau_{ji} = \sigma_j \sigma_i^{-1}$
  end for
Output: the matrix $(\tau_{ji})_{i,j=1}^{m}$ of globally consistent matchings
```

Pachauri D., et al., "Solving the multi-way matching problem by permutation synchronization", Advanced in Neural Information Processing Systems, 2013.

Yu, J.-G., "Globally consistent correspondence of multiple feature sets using proximal Gauss–Seidel relaxation", Pattern Recognition, 2016.

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Step 4. Solve for converged. $\underline{\underline{R}}_S$

Use different algorithm than for the permutations because rotation is 3x3 and that algorithm is less reliable for small matrices.

$$\underline{\underline{R}}_S = \begin{bmatrix} \underline{\underline{R}}_1 \underline{\underline{R}}_1^T & \underline{\underline{R}}_1 \underline{\underline{R}}_2^T & \cdots & \underline{\underline{R}}_1 \underline{\underline{R}}_m^T \\ \underline{\underline{R}}_2 \underline{\underline{R}}_1^T & \underline{\underline{R}}_2 \underline{\underline{R}}_2^T & & \\ \vdots & & \ddots & \\ \underline{\underline{R}}_m \underline{\underline{R}}_1^T & & & \underline{\underline{R}}_m \underline{\underline{R}}_m^T \end{bmatrix} = \begin{bmatrix} \underline{\underline{R}}_1 \\ \underline{\underline{R}}_2 \\ \vdots \\ \underline{\underline{R}}_m \end{bmatrix} \begin{bmatrix} \underline{\underline{R}}_1^T & \underline{\underline{R}}_2^T & \cdots & \underline{\underline{R}}_m^T \end{bmatrix} = \underline{\underline{QQ^T}}$$

$\underline{\underline{R}}_S$ factors into the product of a low rank matrix and its transpose: $\underline{\underline{QQ^T}}$

Instead of optimizing over the space of all 3m x 3m positive semidefinite matrices, search over the space of 3m x 3 matrices. Furthermore, since each $\underline{\underline{R}}_i$ is an orthogonal 3x3 matrix, the vertical concatenation of m of these block matrices ($\underline{\underline{Q}}$) lies in the Stiefel manifold St(p, 3m). Recent research in optimization on Riemannian manifolds allows us to find the solution.

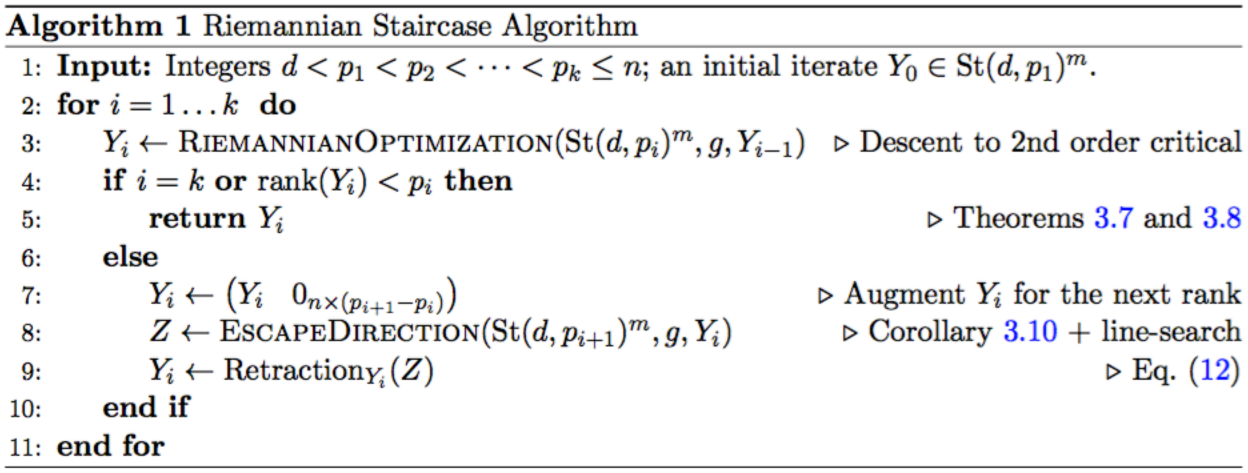
Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Step 4. Solve for converged. $\underline{\underline{R}}_S$

4.1. Starting with an estimate for $\underline{\underline{Q}}$, converge to a local minimum on the Stiefel manifold St(d, 3m)
4.2. Check if this point is a critical point of the semidefinite optimization problem to find $\underline{\underline{R}}_S$
4.3 If it is, terminate the algorithm and project the individual 3x3 blocks in $\underline{\underline{Q}}$ onto their nearest rotation matrix
4.4 Otherwise, increase d by 1 to search a larger Stiefel manifold space St(d+1, 3m) and search again.
4.5 Once a local minimum is found, retract to the nearest St(3, 3m) manifold and go to Step 4.3.

Boumal, N., A. Singer, and P.-A. Absil. *Robust estimation of rotations from relative measurements by maximum likelihood* in *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*, 2013, Firenze, Italy.

Boumal, N., *A Riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints.* http://arxiv.org/abs/1506.00575, 2015.


Algorithm 1 Riemannian Staircase Algorithm

```
1: Input: Integers $d < p_1 < p_2 < \cdots < p_k \leq n$; an initial iterate $Y_0 \in \mathrm{St}(d, p_1)^m$.
2: for $i = 1 \ldots k$ do
3:     $Y_i \leftarrow \mathrm{RiemannianOptimization}(\mathrm{St}(d, p_i)^m, g, Y_{i-1})$   ▷ Descent to 2nd order critical
4:     if $i = k$ or $\mathrm{rank}(Y_i) < p_i$ then
5:         return $Y_i$                                                    ▷ Theorems 3.7 and 3.8
6:     else
7:         $Y_i \leftarrow \begin{pmatrix} Y_i & 0_{n\times(p_{i+1}-p_i)} \end{pmatrix}$   ▷ Augment $Y_i$ for the next rank
8:         $Z \leftarrow \mathrm{EscapeDirection}(\mathrm{St}(d, p_{i+1})^m, g, Y_i)$   ▷ Corollary 3.10 + line-search
9:         $Y_i \leftarrow \mathrm{Retraction}_{Y_i}(Z)$                        ▷ Eq. (12)
10:    end if
11: end for
```

Version 2. "Harder Problem", defined by the fact that $\underline{\underline{X}}$ is unknown.

Current Status:

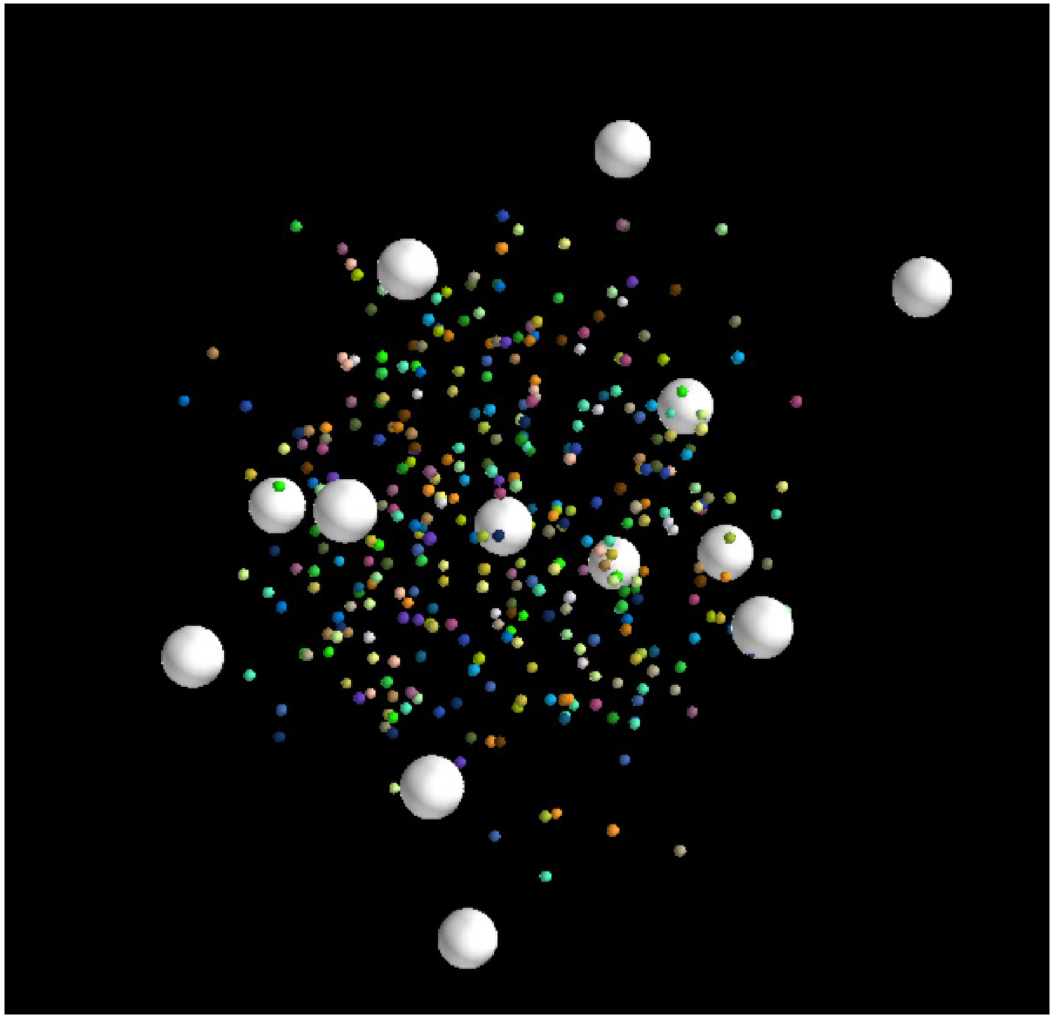
Optimization of Permutation matrices is implemented (in Matlab).
Optimization of Rotation matrices is implemented (in Matlab).

Combined problem is implemented (in Julia).

Works well on artificial data sets.
Dealing with unexpected issues in applications to real data sets.

We think we fixed the bug in the 3D/3D registration solution as of 01.14.16.

Example Problem:



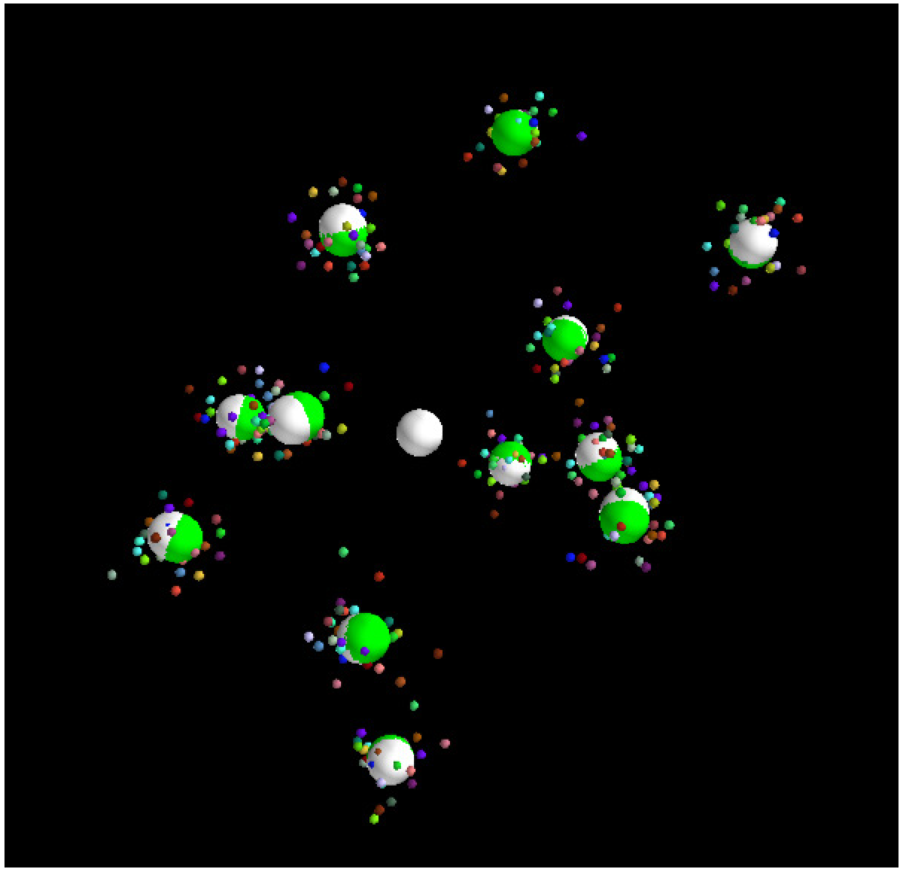
The white atoms represent the reference configuration of N_{ref} = 12 points in three-dimensional space (d=3).

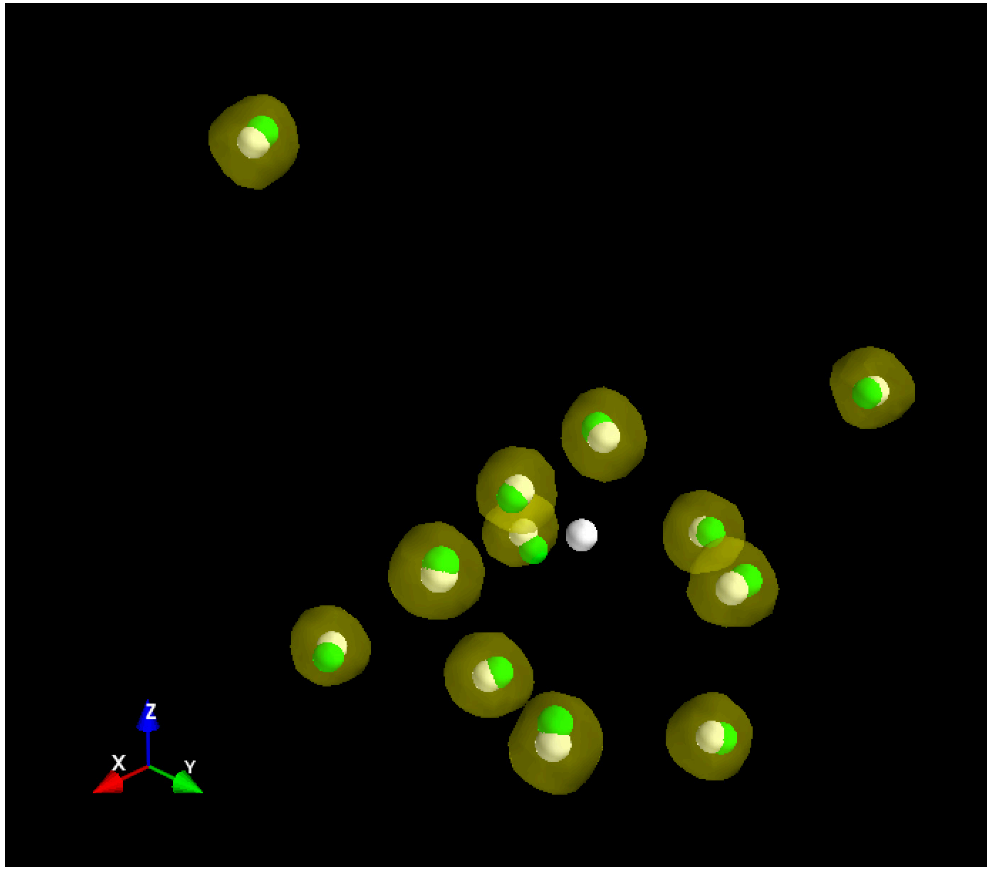
Thirty (m = 30) alternate configurations, each containing 12 points (N_{nbr} = 12), are generated by taking the reference configuration and applying (1) a random permutation, (2) a random rotation and (3) Gaussian noise from the reference position.

To the eye, it is impossible to see any structure in this data set.

d = dimensionality, e.g. d = 3
N_{nbr} = number of points in each configuration, e.g. N_{nbr} = 12
N_{ref} = number of points in reference configuration, e.g. N_{ref} = 12
m = number of configurations, e.g. m = 30 per frame, 1 frame

Example Results:





The large green atoms represent the structure recovered via our optimization procedure.

It is clear that the green atoms are very close to the white atoms.

The yellow clouds represent isocontours for the 30 sets of atoms that were rotated into the optimal, unknown configuration.

Conclusions

An unprecedented “data mining” of atom probe tomography sets can be realized using tools from the field of computer visualization.

We have mapped the problem onto an integrated a set of tools, capable of recovering structure from an APT data set.

There remains work to be done in terms of parallelization, computational efficiency, both in terms of cleaning up code and selecting the best algorithms.

A high resolution three-dimensional experimental data set has never been generated before.

Such a data set opens tremendous doorways in terms of understanding material structure-property relationships.