Matching Paired Sets of Space and Orientation Data

Andrew J. Hanson

hansona@indiana.edu

School of Informatics and Computing

Indiana University

WTC 2018

Champaign-Urbana, IL

16–19 October 2018

Matching Paired Sets of Space and Orientation Data

Why is this interesting ??

Because -

(a) It requires *quaternions* (I wrote the book *Visualizing Quaternions*).

(b) For half a century, **all known solution methods** for the 3D and 4D matching problems have involved numerical approximations. **Exact** algebraic solutions in 3D and 4D were known to involve 4th order polynomials, and, so far as I know, were considered to be **intractable**.

I have used Mathematica to produce the first exact algebraic solutions of the full 3D and 4D eigenspectra ever written down.

BASIC IDEAS of the Matching Problem (aka RMSD)

I: Think about Data Pairs: Protein folding simulations are evaluated by *comparing each result* to a *reference*; Google Map updates involve matching *multiple aerial images*; Satellite positions for GPS *must be tracked* in time and aligned; etc. etc.

II: Task: Optimize a Pair-Matching Measure: The key is to write down a *composite distance measure*, and *rotate* each attempted solution to get as *close as possible* to the reference data.

Outline

- **The 2D Problem — an RMSD Tutorial**

  — Translations, Orientation Frames, 6DOF Data.

- **The 3D Translational (RMSD) problem**

  — Numerical vs Algebraic Quaternion Eigenvalue solutions

- **The 3D Orientation problem (QRMSD)**

- **The 6DOF problem (RMSD+QRMSD)**

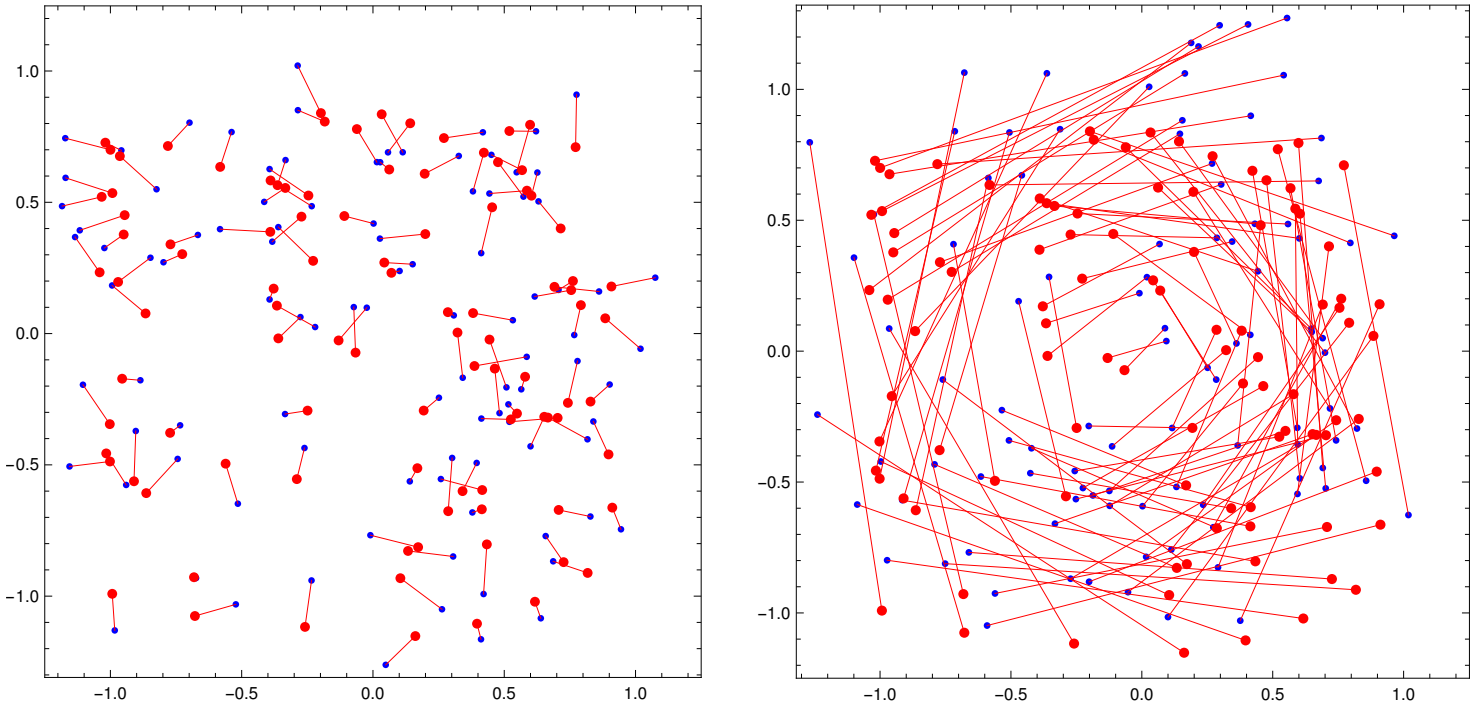
RMSD in Two Dimensions

- Our input data are the *reference* data set $\{y_k\}$ and one or more item-by-item matched *test* data sets $\{x_k\}$.

- The task (in any dimension D) is to **PICK ONE ROTATION MATRIX** $\boxed{R_{\mathrm{D}}}$ that minimizes the *Root-Mean-Square Deviation* measure

$$\mathrm{RMSD_D}^2 \rightarrow \mathrm{S_D}^2 = \sum_{k=1}^{N} \|R_{\mathrm{D}} \cdot x_k - y_k\|^2$$

Example: Collections of $\|x_k - y_k\|^2$



Left: *Reference* data $\{y_k\}$ are in red, noisy *test* data $\{x_k\}$ in blue, *before* global rotation. **Right:** What happens to the incremental distances after a **global rotation** of the noisy blue test data around the mutual center of mass.

Refining the 2D Problem

Method: Minimizing the RMSD is equivalent (dropping constants) to *maximizing* the simpler cross-term measure Δ:

$$\Delta_2(\theta) = \sum_{k=1}^{N} (R_2(\theta) \cdot x_k) \cdot y_k = \sum_{a=1,b=1}^{2} {R_2}^{ba} E_{ab} \, ,$$

where the 2D rotation matrix is $R_2(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$, and the "data contents" all reduce (in any dimension) to Euclidean component averages,

$$E_{ab} = \sum_{k=1}^{N} x_k^a y_k^b \, .$$

The raw form of the 2D spatial RMSD optimization task is thus to find the θ_0 producing the $R_2(\theta_0)$ that maximizes Δ_2.

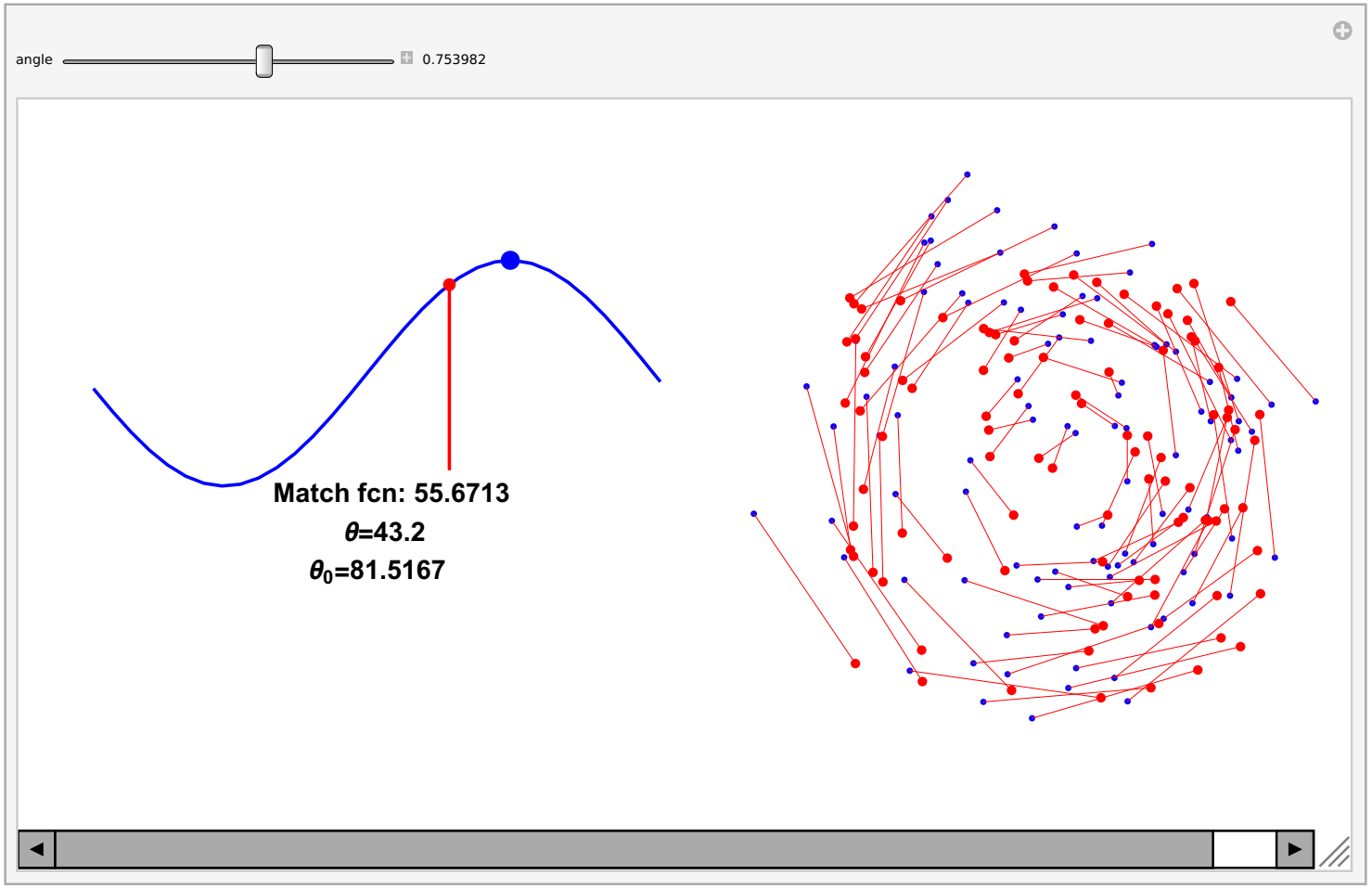
Solving 2D Spatial RMSD is Easy

We can either differentiate with respect to θ and set $\Delta_2'(\theta) = 0$, or simply observe directly that $\Delta_2(\theta)$ is largest when the vector $(\cos\theta, \sin\theta)$ is parallel to its coefficients, both arguments leading to the solution

$$\tan\theta_0 = \frac{E_{12} - E_{21}}{E_{11} + E_{22}} \equiv \frac{N}{M}$$

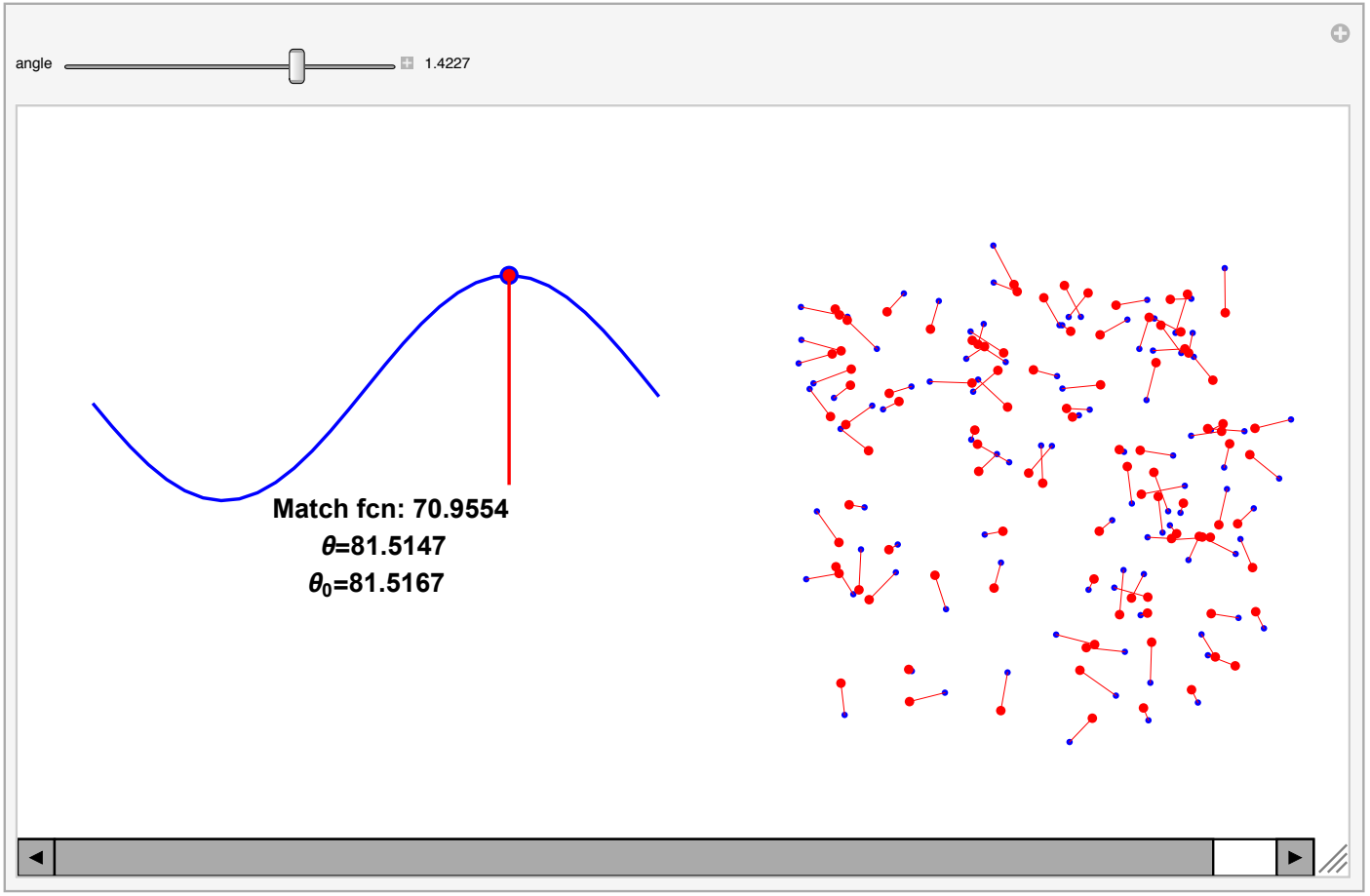
$$(\cos\theta_0, \sin\theta_0) = \left(\frac{M}{\sqrt{M^2 + N^2}}, \frac{N}{\sqrt{M^2 + N^2}}\right)$$

Solving – how the optimization looks!



Partway through the rotation from mismatched state, still lots of space between the blue and red points.

At the Solution!



$$R_2(\theta_0) = R_2\left(\arctan\left(\frac{E_{12} - E_{21}}{E_{11} + E_{22}}\right)\right)$$

rotates the blue points to optimal alignment with red points.

2D Demo

A single rotation matrix moves the test *blue* points back and forth, reaching the *closest* "RMSD" alignment at the angle

$$\theta_0 = \arctan\left(\frac{E_{12} - E_{21}}{E_{11} + E_{22}}\right)$$

that we solved for *algebraically*!

Not So Fast!

This was too easy: This θ_0 solution works only in 2D! For a method that extends to 3D, we must replace $\Delta'(\theta) = 0$ by a quaternion-compatible linear algebra problem.

Weirdly, this is easy if we replace the angle θ by its *half-angles*, that is

$\cos\theta = a^2 - b^2$	$\sin\theta = 2ab$
$a = \cos\frac{\theta}{2}$	$b = \sin\frac{\theta}{2}$

.

So

$$R_2(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} a^2 - b^2 & -2ab \\ 2ab & a^2 - b^2 \end{bmatrix}$$

Amazing fact: This lets us *break apart* the Δ_2 optimization function into a *matrix eigenvalue problem*.

2D RMSD as an Eigensystem

Alternative, generalizable, approach: Start with the half-angle unit vector (a, b) and *quadratic form* for the rotation matrix $R_2(\theta) = R_2(a, b)$. Then

$$
\begin{aligned}
\tilde{\Delta}_2(a,b) &= \mathrm{tr}\left(\begin{bmatrix} a^2-b^2 & -2ab \\ 2ab & a^2-b^2 \end{bmatrix} \cdot \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}\right) \\
&= a^2\,(E_{11}+E_{22}) - b^2\,(E_{11}+E_{22}) \\
&\quad + 2ab\,(E_{12}-E_{21}) \\
&= [a\ b] \begin{bmatrix} E_{11}+E_{22} & E_{12}-E_{21} \\ E_{12}-E_{21} & -(E_{11}+E_{22}) \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \\
&= [a\ b] \begin{bmatrix} M & N \\ N & -M \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \mathbf{a}\cdot\mathbf{M}\cdot\mathbf{a}\ .
\end{aligned}
$$

2D Eigensystem, contd

Key new element is the traceless, symmetric "profile" matrix $\mathbf{M}$, built of data components

$$M = E_{11} + E_{22} \quad \textit{and} \quad N = E_{12} - E_{21}$$

with eigenvalues $\lambda_\pm = \pm\sqrt{M^2 + N^2}$. From standard linear algebra, the maximal value of $\tilde{\Delta}_2(a, b)$ occurs when $\mathbf{a} = (a_0, b_0)$ is the eigenvector of λ_+. These are *exactly* the half-angle objects corresponding to our earlier $\theta_0 = \arctan(N/M)$!

$$\begin{aligned} a_0 &= \cos(\theta_0/2) = \sqrt{\tfrac{\lambda+M}{2\lambda}} \\ b_0 &= \sin(\theta_0/2) = \mathrm{sign}(N)\sqrt{\tfrac{\lambda-M}{2\lambda}} \,. \end{aligned}$$

No, the $\mathrm{sign}(N)$ *is not a typo...*

2D Spatial RMSD summary

The solutions $\cos\theta_0 = M/\lambda$ and $\sin\theta_0 = N/\lambda$, from direct maximization of $\Delta_2(\theta)$, are **the same** as the solutions

$$a_0{}^2 - b_0{}^2 \quad \textit{and} \quad 2a_0b_0$$

from the eigenvector (a_0, b_0) of the data matrix $\mathbf{M}$.

This technique extends to 3D, using quaternions q instead of (a, b), and is the basis of the conventional numerical method for finding the 3D rotation that minimizes the 3D RMSD.

We will write down an analogous but previously unknown, nontrivial, algebraic solution to the 3D translational RMSD matching problem.

Remark: **Quaternion Frames: the QRMSD Problem**

The *spatial* matching problem can be generalized to *orientation frames*.

Why is this interesting? Example: Proteins contain hundreds of amino acid residues, and the unique orientations of each residue can be encoded efficiently as **QUATERNION FRAMES**.

We have also solved the closed form **quaternion frame matching problem**, aka the **QRMSD** problem, using our methods.

Finally, the 3D Spatial RMSD Problem

The 2D exercises we have done up until now are really useful for seeing just how the geometry of the RMSD problem works out.

- We will now address the **main problem** of finding the optimal 3D spatial rotation needed to match two 3D data sets.
- It was discovered independently, in three different literatures, that this problem reduces to a 4D **quaternion eigenvalue problem**, and finding the numerical solution is much easier in that framework.
- **We now show how to use Mathematica to completely solve this 4D problem, in principle a very difficult quartic algebraic equation, analytically.**

Summarize Quaternion Properties

- **Unit four-vector.** Take $q = (q_0, q_1, q_2, q_3) = (q_0, \vec{\mathbf{q}})$ to obey the constraint $q \cdot q = 1$.

- **Multiplication rule.** The quaternion product of q and p is
  $q * p = (q_0 p_0 - \vec{\mathbf{q}} \cdot \vec{\mathbf{p}},\ q_0 \vec{\mathbf{p}} + p_0 \vec{\mathbf{q}} + \vec{\mathbf{q}} \times \vec{\mathbf{p}})$,
  or, alternatively,

$$\begin{bmatrix} [q*p]_0 \\ [q*p]_1 \\ [q*p]_2 \\ [q*p]_3 \end{bmatrix} = \begin{bmatrix} q_0p_0 - q_1p_1 - q_2p_2 - q_3p_3 \\ q_0p_1 + q_1p_0 + q_2p_3 - q_3p_2 \\ q_0p_2 + q_2p_0 + q_3p_1 - q_1p_3 \\ q_0p_3 + q_3p_0 + q_1p_2 - q_2p_1 \end{bmatrix}$$

Exact Quaternion Soln to 3D RMSD

The 3D RMSD problem looks very similar to 2D, with 9 data averages instead of 4:

$$E_{ab} = \sum_{k=1}^{N} x_k^a y_k^b \ .$$

The trick is to re-express the minimization function Δ_3 in terms of the sought-for rotation R(quaternion)

$$\Delta_3(q) = \sum_{a,b} R^{ba}(q) E_{ab} = \sum_{i,j=0}^{3} q_i M_{ij} q_j \ .$$

Quaternion Part of 3D RMSD

This last expression is the 3D equivalent of using our 2D matrix $R(a,b)$ to break up the 3D rotation, which has the quaternion quadratic form

$$R(q) = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2q_1q_2-2q_0q_3 & 2q_1q_3+2q_0q_2 \\ 2q_1q_2+2q_0q_3 & q_0^2-q_1^2+q_2^2-q_3^2 & 2q_2q_3-2q_0q_1 \\ 2q_1q_3-2q_0q_2 & 2q_2q_3+2q_0q_1 & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix}.$$

The decomposition $\Delta \rightarrow \Delta(q)$ is then a matrix form

$$\Delta_3(q) = \sum_{i,j=0}^{3} q_i \mathrm{M}_{ij} q_j$$

where M is a novel linear combination defined as . . .

The 3D RMSD Profile Matrix $M_3(E)$ Is Four-Dimensional!

$$\begin{bmatrix} E_{xx}+E_{yy}+E_{zz} & E_{yz}-E_{zy} & E_{zx}-E_{xz} & E_{xy}-E_{yx} \\ E_{yz}-E_{zy} & E_{xx}-E_{yy}-E_{zz} & E_{xy}+E_{yx} & E_{zx}+E_{xz} \\ E_{zx}-E_{xz} & E_{xy}+E_{yx} & -E_{xx}+E_{yy}-E_{zz} & E_{yz}+E_{zy} \\ E_{xy}-E_{yx} & E_{zx}+E_{xz} & E_{yz}+E_{zy} & -E_{xx}-E_{yy}+E_{zz} \end{bmatrix}$$

Observe that this matrix is **traceless and symmetric**, which simplifies some aspects of the 3D problem.

(It turns out, we can also solve the **4D** Euclidean RMSD problem analytically, where the matrix M is completely general, with no constraints whatever.)

Eigenvalues of the Profile Matrix M

The maximum value of

$$\Delta_3 = \mathrm{tr}\,(R_3 \cdot E) = q \cdot M_3 \cdot q$$

is known from linear algebra to occur for the *maximal* eigenvalue (i.e., the quaternion eigenvector q_{opt} corresponding to θ_0 in the 2D case).

We know we must solve $\det[M - eI_4] = 0$, where e denotes a generic eigenvalue and I_4 is the 4D identity matrix. We can write this in two ways:

$$e^4 + e^3 p_1 + e^2 p_2 + e p_3 + p_4 = 0\ ,$$

$$(e - \epsilon_1)(e - \epsilon_2)(e - \epsilon_3)(e - \epsilon_4) = 0\ .$$

where ϵ_i are the eigenvalues, $p_1 = 0$ is the trace, $p_4(E)$ is the determinant, and $p_2(E)$ and $p_3(E)$ are polynomials in the data averages E_{ab}.

Eigenvalues of the Profile Matrix **M**

Observe the following:

- The data coefficients $\{p_1(E), p_2(E), p_3(E), p_4(E)\}$ are known: *they are just numbers.*

- The equations above give us the *unknown eigenvalues* ϵ_i in terms of the *known data coefficients:*

$$
\begin{aligned}
p_1(E) &= -\epsilon_1 - \epsilon_2 - \epsilon_3 - \epsilon_4 \\
p_2(E) &= \epsilon_1\epsilon_2 + \epsilon_1\epsilon_3 + \epsilon_2\epsilon_3 + \epsilon_1\epsilon_4 + \epsilon_2\epsilon_4 + \epsilon_3\epsilon_4 \\
p_3(E) &= -\epsilon_1\epsilon_2\epsilon_3 - \epsilon_1\epsilon_2\epsilon_4 - \epsilon_1\epsilon_3\epsilon_4 - \epsilon_2\epsilon_3\epsilon_4 \\
p_4(E) &= \epsilon_1\epsilon_2\epsilon_3\epsilon_4
\end{aligned}
$$

- **Therefore our task is to invert this equation for** $\epsilon_i(p_1, p_2, p_3, p_4)$**.**

Approach to Exact Quaternion Soln

Studying the apparently intractable algebraic expressions for $\mathbb{M}$'s numerical solutions in Mathematica over a period of a year, I discovered that it was useful to *change the variables* in the above (traceless) equation to express the 4 eigenvalues of $\mathbb{M}$ in the following form:

$$\begin{aligned}
\epsilon_1 &= +\sqrt{X} + \sqrt{Y} + \sqrt{Z} \\
\epsilon_2 &= +\sqrt{X} - \sqrt{Y} - \sqrt{Z} \\
\epsilon_3 &= -\sqrt{X} + \sqrt{Y} - \sqrt{Z} \\
\epsilon_4 &= -\sqrt{X} - \sqrt{Y} + \sqrt{Z}
\end{aligned}$$

This is a peculiarly advantageous form of the 4D symmetric, traceless matrix $\mathbb{M}$ that is adapted to machine algebra.

Exact Quaternion Soln to 3D RMSD

The equations for the eigenvalues reduce to this form, where of course *knowing* $(X(p), Y(p), Z(p))$ *means knowing the eigenvalues* ϵ_i*:*

$$\text{Eqns} = \left\{ \begin{array}{rcl} p_4(E) & = & X^2 - 2XY + Y^2 - 2XZ - 2YZ + Z^2 \\ [p_3(E)]^2 & = & 64\,XYZ \\ p_2(E) & = & -2(X + Y + Z) \end{array} \right\}$$

Then `Solve[ Eqns,{X, Y, Z}]` runs for a minute, giving 6 sets of solutions with this `ByteCount` list:

{{4584, 19544, 21552}, {4584, 19448, 21552}, {5224, 22232, 28640},
{5224, 22400, 28736}, {5224, 22256, 28632}, {5224, 22392, 28728}}

Exact Quaternion Soln to 3D RMSD, contd

Each of the *short* expressions looks like this, e.g., for the first $X(p)$, while the other expressions are pages of algebra that do not respond to `Simplify[]` :

$$-\frac{p_2}{6}$$

$$-\frac{\sqrt[3]{-p_2^3+36p_4p_2+\frac{3}{2}\left(\sqrt{-48p_4p_2^4+12p_3^2p_2^3+384p_4^2p_2^2-432p_3^2p_4p_2+81p_3^4-768p_4^3}-9p_3^2\right)}}{12}$$

$$-\frac{p_2^2+12p_4}{12\sqrt[3]{-p_3^3+36p_2p_4+\frac{3}{2}\left(\sqrt{-48p_4p_2^4+12p_3^2p_2^3+384p_4^2p_2^2-432p_3^2p_4p_2+81p_3^4-768p_4^3}-9p_3^2\right)}}$$

Note: Why you need analytic solns:

Another basic problem is that you can't include any parameters: if m_0 and m_1 are 4 $\times$ 4 profile matrices, you get abstract roots:

$$\text{Eigenvalues}[t_0 m_0 + t_1 m_1] \Longrightarrow$$

```
{Root[-275457. t0^4 - 1.24918*10^6 t0^3 t1 - 1.98623*10^6 t0^2 t1^2 -
    1.33246*10^6 t0 t1^3 -  319975. t1^4 + (-47234.5 t0^3 - 149423. t0^2
       153487. t0 t1^2 - 51093.6 t1^3) #1 + (-2131.46 t0^2 -
       4172.69 t0 t1 - 2147.62 t1^2) #1^2 - 1.77636*10^-15 t0 #1^3 +
         #1^4 \&, 1],
 Root[-275457. t0^4 - 1.24918*10^6 t0^3 t1 - 1.98623*10^6 t0^2 t1^2 -
   1.33246*10^6 t0 t1^3  ...   #1^4 \&, 2],
  Root[ ... ...   #1^4 \&, 3],
  Root[ ... ...   #1^4 \&, 4] }
```

Exact Quaternion Soln to 3D RMSD

So *what happens* is that 2/3 of every solution is pages of impenetrable symbols, 20Kb each, but the FIRST one is only 5Kb, as on the previous slide.

If you plug in random numbers, you find that each of the short ones matches one of the long ones, so if you assemble them like a puzzle, you have a completely usable algebraic solution.

THAT solution in turn breaks magically into a sum of **cube roots of unity**, hugely simplifying the whole thing.

The Solution

By comparing the numerical values to apparently intractable algebraic expressions using the tricks above, I was ultimately able to reduce the algebra for the **all eigenvalues of** M to the following form for $(X(p), Y(p), Z(p)) = F_{(x,y,z)}$:

$$F_f(p_2, p_3, p_4) \;=\; \tfrac{1}{6}\left(r(p_2, p_3, p_4)\,\mathrm{cos}_f(p_2, p_3, p_4) - p_2\right)$$

where we define

$$\begin{aligned}
\mathrm{cos}_x(p_2, p_3, p_4) &= \mathrm{cos}\left(\frac{\mathrm{arg}(a+i\,b)}{3}\right) \\
\mathrm{cos}_y(p_2, p_3, p_4) &= \mathrm{cos}\left(\frac{\mathrm{arg}(a+i\,b)}{3} - \frac{2\pi}{3}\right) \\
\mathrm{cos}_z(p_2, p_3, p_4) &= \mathrm{cos}\left(\frac{\mathrm{arg}(a+i\,b)}{3} + \frac{2\pi}{3}\right)
\end{aligned}$$

Exact Quaternion Soln to 3D RMSD

In this expression $\arg(u+i\,v) = \mathrm{atan2}(v,u) = \mathrm{ArcTan}(u,v)$, $F_f(p)$ corresponds to $X(p)$, $Y(p)$, and $Z(p)$ for $f = \{x, y, z\}$, and the utility functions reduce to

$$
\begin{aligned}
a(p_2,p_3,p_4) &= {p_2}^3 + \tfrac{1}{2}\left(27{p_3}^2 - 72p_2p_4\right) \\
n(p_2,p_3,p_4) &= {p_2}^2 + 12p_4 \\
b(p_2,p_3,p_4) &= \sqrt{n^3 - a^2} \\
r(p_2,p_3,p_4) &= \sqrt[6]{a^2 + b^2} \;=\; \sqrt{n}\,.
\end{aligned}
$$

Exact Quaternion Soln to 3D RMSD

The (all real) 3D eigenvalues in order of descending magnitude are now written in terms of the three phases of $F_f(p)$ for $f = \{x, y, z\}$ corresponding to $\{X, Y, Z\}$:

$$\begin{aligned}
\epsilon_1 &= +\sqrt{X} + \sqrt{Y} + \sqrt{Z} \\
\epsilon_2 &= +\sqrt{X} - \sqrt{Y} - \sqrt{Z} \\
\epsilon_3 &= -\sqrt{X} + \sqrt{Y} - \sqrt{Z} \\
\epsilon_4 &= -\sqrt{X} - \sqrt{Y} + \sqrt{Z} \; .
\end{aligned}$$

Eigenvectors for 3D RMSD

The eigenvector formulas corresponding to ϵ_k can be generically computed by solving the bottom three rows of

$$[M_3 \cdot \mathbf{v} - e\mathbf{v}] = 0$$

for the elements of $\mathbf{v} = (1, v_1, v_2, v_3)$ as a function of some eigenvalue e, so we just use the exact algebraic solution for $e = \epsilon_1$ and we are done!

Caveat - rearrange $\mathbf{v}$ if any element of M is already diagonal!

Exact Quaternion Soln to 3D RMSD

- **This solution is unknown to the best of our knowledge.** All previous literature solves *numerically* using, e.g., Newton's method, for the maximal quaternion eigenvalue, giving the quaternion solution for the optimal aligning rotation via the quadratic formula for $R_3(q)$. In fact, we get **all four** eigenvalues, not just the maximal one.

- **Using similar methods, we have solved the** *3D frame alignment problem* **in closed form as well.**

- **Using still another tricky algebraic manipulation,** there is also a way to solve the full 6 DOF alignment problem with one *mandatory arbitrary dimensional parameter* needed to rescale the distance measure in Length units to the frame-alignment measure in Radians.

Outline: Generalizing to 3D Quaternion Frames

The *spatial* matching problem can be generalized to a *frame orientation* problem.

Why is this interesting? Example: Proteins contain hundreds of amino acid residues, and the unique orientations of each residue can be encoded efficiently as **Quaternion Frames**.

We next briefly outline the closed form **quaternion frame matching** aka **QRMSD** problem.

Quaternions as 3D Frames

If we take a quaternion q and write the 3D rotation matrix as

$$R(q) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

the *columns* are the axes of a 3D orthonormal frame triad.

Taking a set of almost-aligned quaternion-valued test frames $\{p_k\}$ and reference frames $\{r_k\}$, we can write the chord-distance measure as

$$\Delta_f = \sum_{k=1}^{N} (q * p_k) \cdot r_k \ ,$$

where the quaternion product of q and p is

$$q * p = (q_0 p_0 - \vec{\mathbf{q}} \cdot \vec{\mathbf{p}},\ q_0 \vec{\mathbf{p}} + p_0 \vec{\mathbf{q}} + \vec{\mathbf{q}} \times \vec{\mathbf{p}}),$$

. . . Solution for 3D Frames

Rearrange in Quaternion Form:

We see the quaternion analog of the *profile matrix* emerge as $E_{ab} = \sum_{k=1}^{N} p_k^{(a)} r_k^{(b)}$, and

$$\Delta_f = q \cdot V$$

where V is the column vector

$$V = \begin{bmatrix} E_{00} + E_{11} + E_{22} + E_{33} \\ -E_{01} + E_{10} - E_{23} + E_{32} \\ -E_{02} + E_{20} - E_{31} + E_{13} \\ -E_{03} + E_{30} - E_{12} + E_{21} \end{bmatrix}$$

exactly analogous to the spatial *profile matrix*.

. . . Exact Solution for 3D Frames

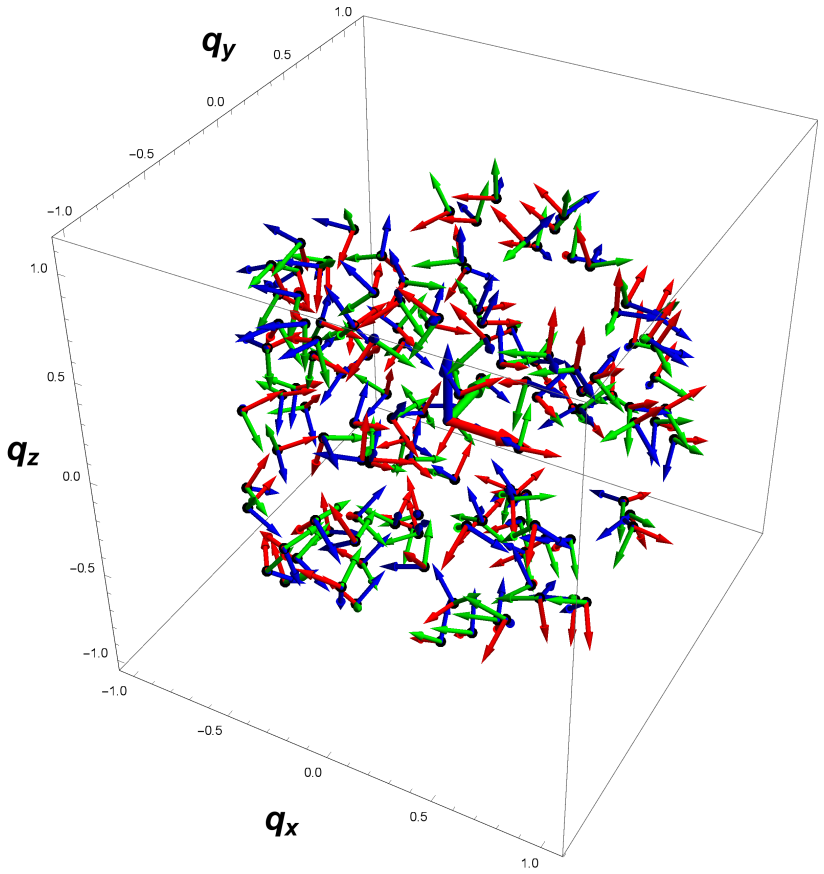
The exact solution (assuming resolved quaternion sign ambiguities in the data) is just

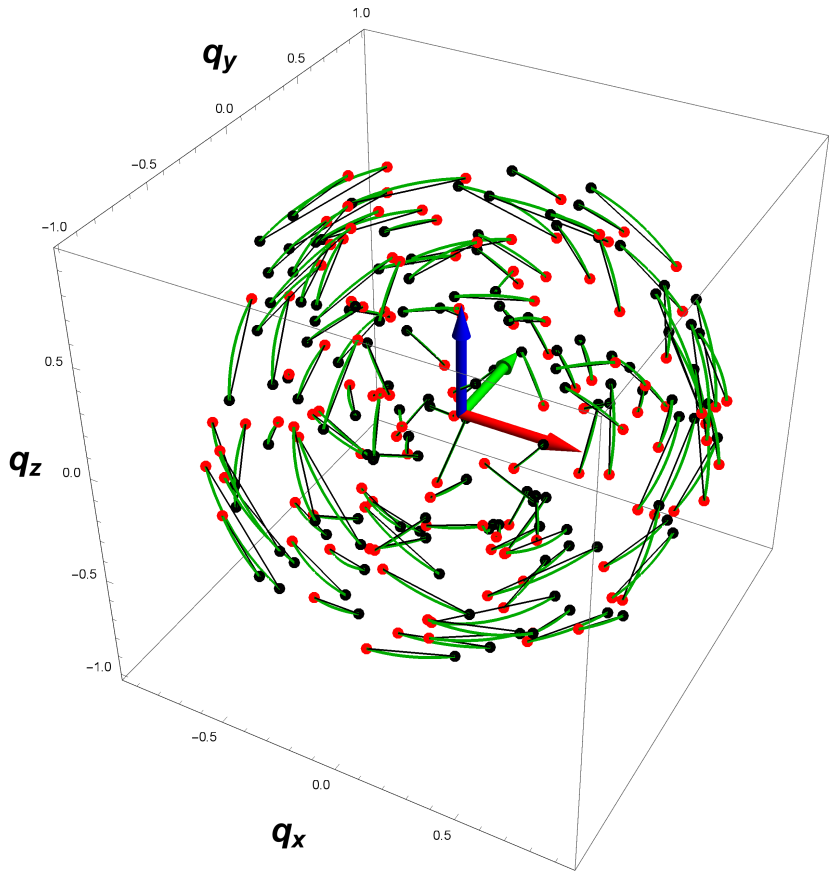
$$q_0 = \frac{V}{\|V\|}$$

and the "cost" of the alignment at the optimal q_0 is therefore very simple:

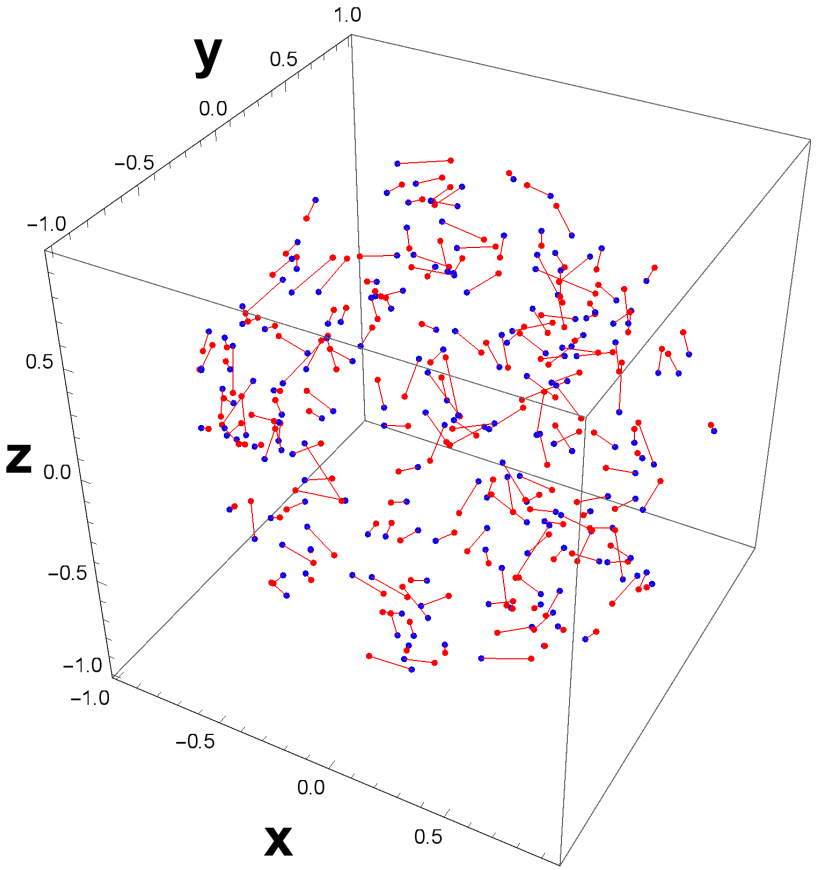
$$\Delta_f(q_0) = q_0 \cdot V = \|V\| \ .$$

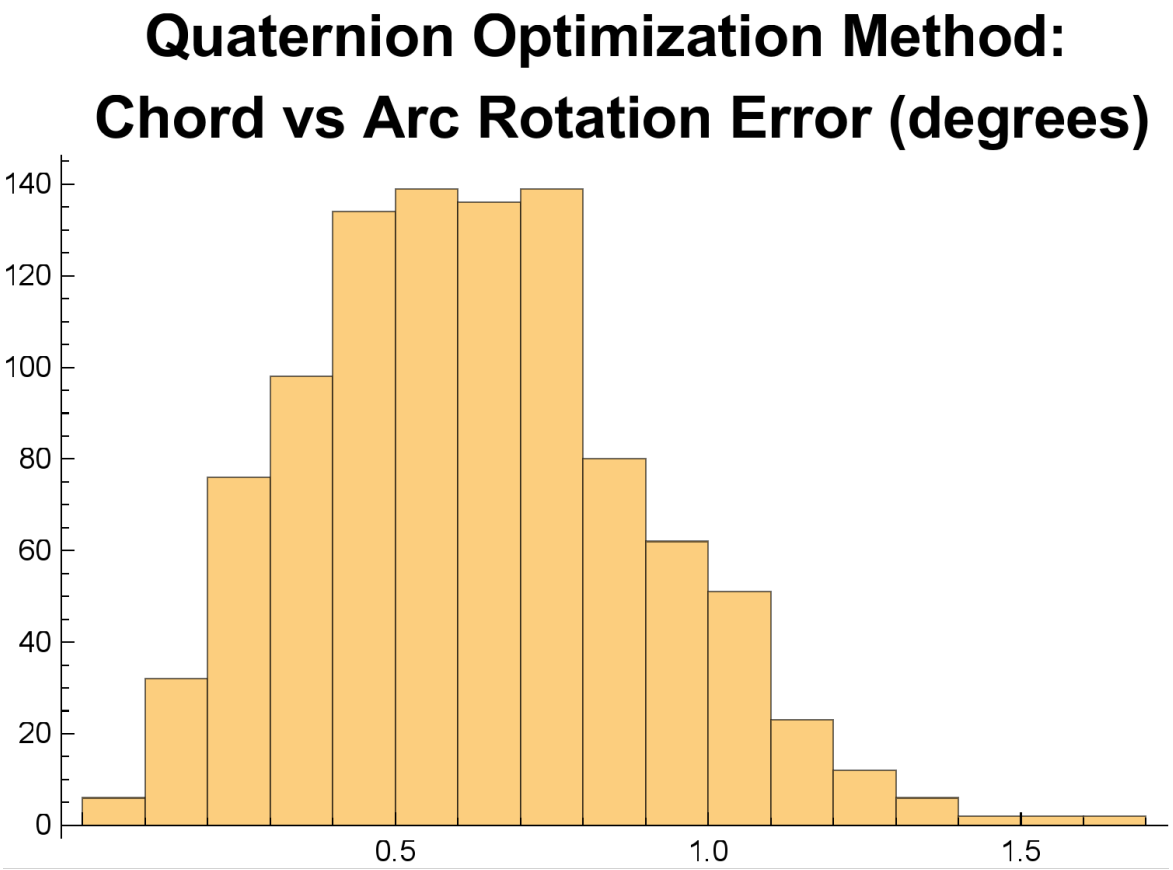
Slides of Quaternion Frame Analysis





Slides of Quaternion Frame Analysis, contd.





Remark on the 4D Data Problem

It turns out that while 3D data, with a 3×3 data matrix E_{3ab}, can be optimized with a quaternion-based symmetric, traceless 4×4 profile matrix $\mathrm{M}(E_3)$, a slight extension of the 3D quaternion method expresses the 4D Euclidean data problem, with a 4×4 data matrix E_{4ab}, in terms of a quaternion eigenvalue problem with a *non-traceless, non-symmetric* 4×4 profile matrix $\mathrm{M}(E_4)$.

We have an extension of the method given here that solves both the 4D spatial data and the 4D quaternion frame data problems *exactly*.

Summary

- **Defined RMSD Spatial Matching Problem.** Find a rotation that best aligns a pair of matched lists of spatial, orientation frame, or combined data. Used the 2D system to build intuition.

- **Presented the exact solutions of both spatial and frame data matching for the 3D problem.** The expression in terms of a triple of cube roots of unity was unknown and unexpected.

- **Also solved the 4D spatial and orientation frame RMSD problems in terms of exact algebraic eigenvalues of completely general 4D real matrices.**

Blank Slide

- .
- .

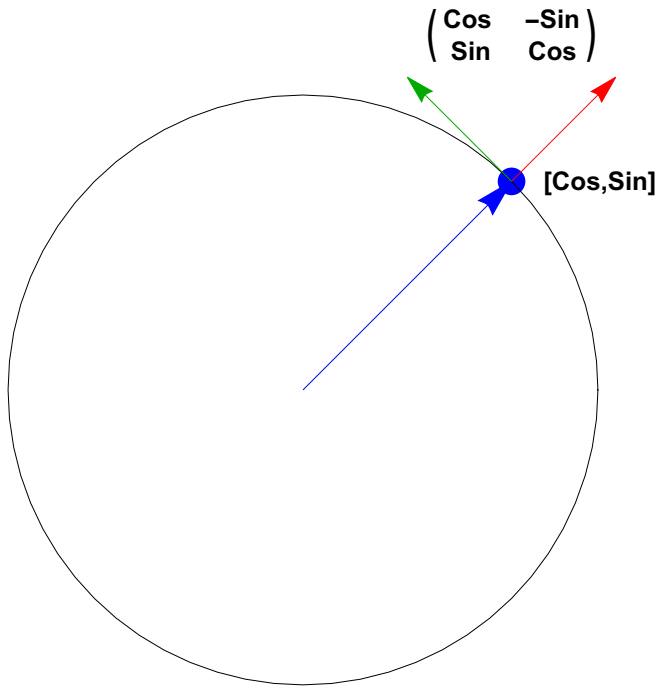
REPEAT with 2D Orientation Frames

Orientation Frames can be adjoined to each point (this happens for amino acids in proteins, for example). Since they also can be misaligned by a global rotation, we can repeat the entire matching procedure for orientation: first in 2D, then extending to **quaternion-based frame representations** in 3D.

A 2D frame can be represented as a single unit-norm complex number $e^{i\theta} = \cos\theta + i\sin\theta$, interpreted as a pair of orthonormal vectors representing the frame:

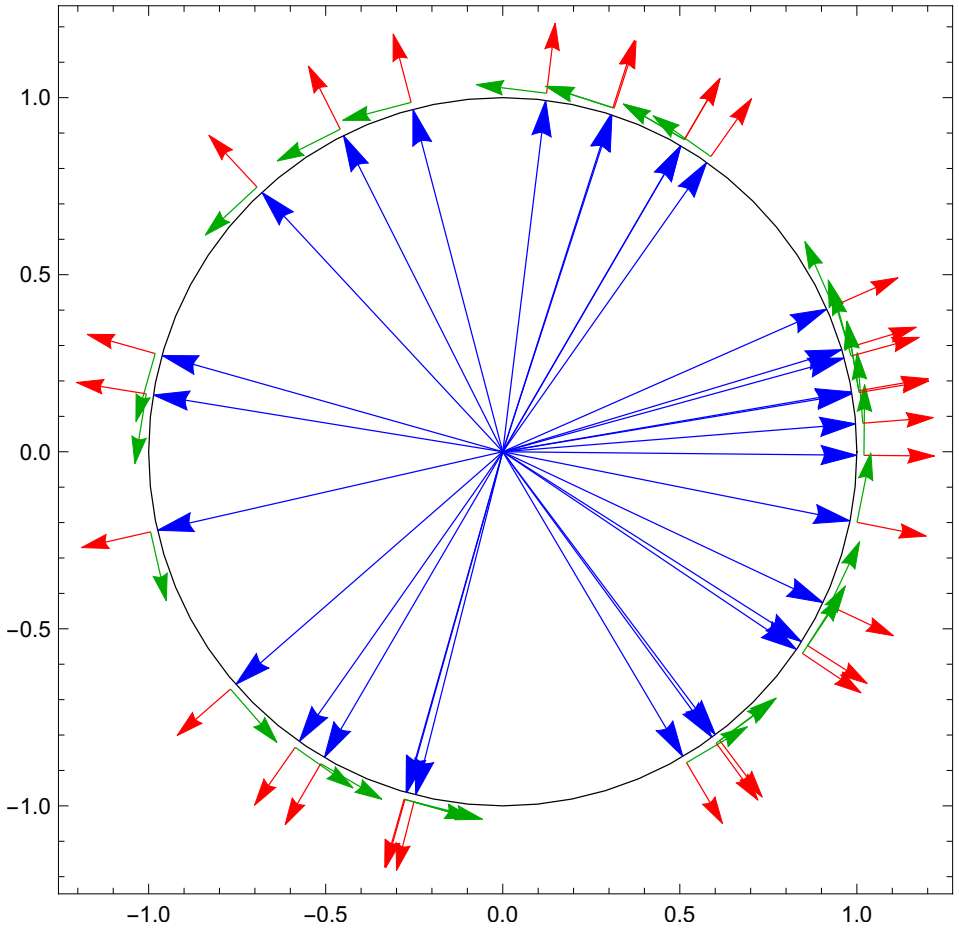
$$\begin{aligned} \hat{\mathbf{u}} &= (\cos\theta, \sin\theta) \\ \hat{\mathbf{v}} &= (-\sin\theta, \cos\theta) \ . \end{aligned}$$

Visualizing a 2D Frame



In 2D, an angle θ represents an orthonormal frame (u, v).

Visualizing 2D Orientation Collections



Our 2D frame data sets can then be represented as points on the unit circle.

Simplest version: 2D Frame Matching

Suppose we have a data set with a test set of 2D coordinate frames represented by angles $\{\alpha_k\}$ to be compared with a reference set of frame angles $\{\beta_k\}$. A close analog of the translational cumulative measure of the distances between frames is just the sum of squared angular distances

$$S_f{}^2 = \sum_{k=1}^{N} (\alpha_k - \beta_k)^2 \ .$$

2D Frame Minimizer

Since rotating a 2D frame (represented by α_k) by an angle θ is just complex multiplication of the form

$$e^{i\theta} e^{i\alpha_k} = e^{i(\theta+\alpha_k)} ,$$

a rotation of the entire set of test frames by the same angle has a cumulative measure of

$$\begin{aligned} {S_f}^2(\theta) &= \sum_{k=1}^{N} (\theta + \alpha_k - \beta_k)^2 \\ &= N\theta^2 + 2\theta \sum_{k=1}^{N} (\alpha_k - \beta_k) + \text{const} . \end{aligned}$$

. . . minimizing 2D frame match

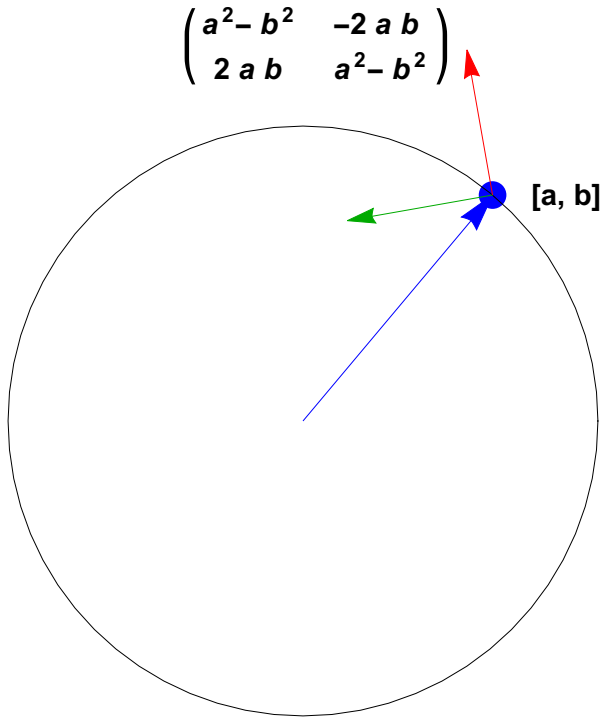
Differentiating with respect to θ, we see the optimal angle is essentially the center of mass of the differences,

$$\theta_0 = -\frac{1}{N} \sum_{k=1}^{N} (\alpha_k - \beta_k) \ .$$

This news is too good to be true: only in 2D do the angular variables behave *exactly* like a linear Euclidean space.

Nothing like this appears in higher dimensions.

As before: Need a double-valued 2D Frame



Extendability to 3D requires *half-angles*:

. . . double-valued 2D Frame

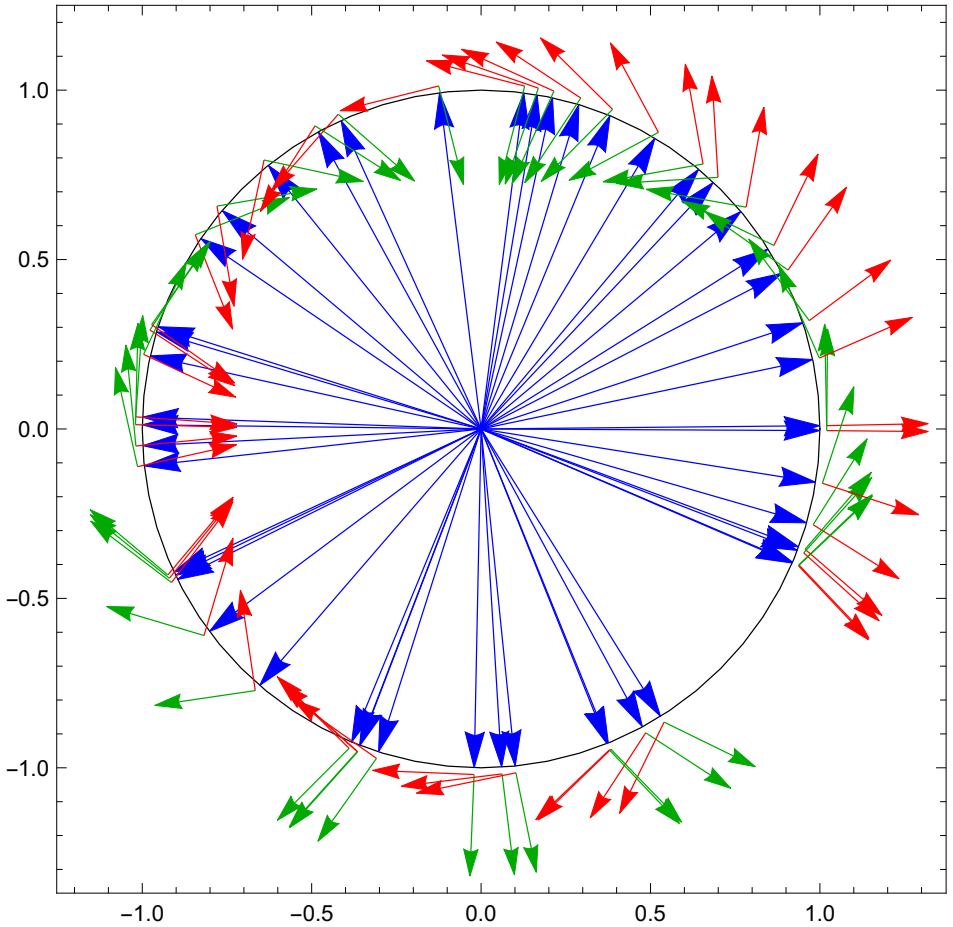
That is

$$(a, b) = (\cos(\theta/2), \sin(\theta/2)) \ ,$$

so the frame denoted by $(a, b) = e^{i\theta/2}$ is our old friend

$$\begin{bmatrix} a^2 - b^2 & -2ab \\ 2ab & a^2 - b^2 \end{bmatrix} \ .$$

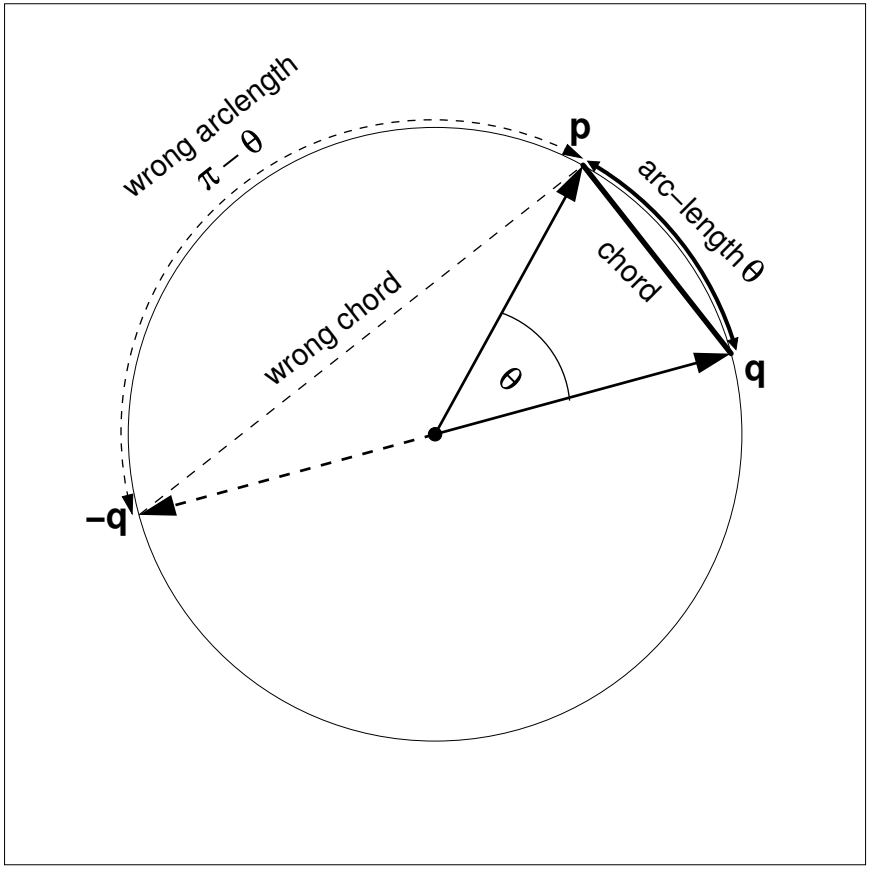
Visualizing 2D Orientation Collections



In 2D quaternions, the points (a, b) and $(-a, -b)$ on the unit circle *now* represent ONE orthonormal frame

$$u = (a^2 - b^2, 2ab),\ v = (-2ab, a^2 - b^2).$$

Tricky Part of Quaternion Distances



Because of sign ambiguity – the vectors q and $-q$ give the same rotation matrix — you need to choose $|\cos\theta|$ or the *minimum* of the chord distances to get the *right* arc-length or chord measure.

Frame distance with 2D chords

The **chord distance** on the previous slide is in fact the key to making the angular distance work like the spatial distance.
It is an **approximation**, but it is a very good approximation.
If we simply use $\| (R_2(\theta/2) \cdot \mathbf{p}) - \mathbf{r} \|$ as the fundamental minimizing measure, where $\mathbf{p}$ and $\mathbf{r}$ are unit vectors in the 2D complex plane, the optimal maximizing chord distance can be written

$$\Delta_f = \sum_{k=1}^{N} \cos\left(\theta/2 + b_k/2 - c_k/2\right)$$

Here $\mathbf{p} = (\cos(b/2), \sin(b/2))$, etc., to be quaternion-like.

Frame distance with 2D chords

After some manipulation with trigonometric arithmetic, the measure Δ_f can be written in essentially the same form as the Euclidean measure,

$$\Delta_f(a,b) = a(\theta)(E_{00}+E_{11})+b(\theta)(E_{01}-E_{10}) = a\tilde{M}+b\tilde{N},$$

where $E_{ij} = \Sigma_{k=1}^{N} {p_k}^i {r_k}^j$. This is a linear equation in (a,b) and can be solved exactly as before.

Remarkably, this situation repeats in exactly the same way for the closed form solution of the 3D quaternion frame problem with the chord-measure approximation.

Space+Frame distance

Note: In fact, if one uses $\left(\tilde{\Delta}_f\right)^2$ as a measure, the angular optimization problem can be written **in terms of a quadratic form in** (a, b), and thus can be folded directly into the **spatial optimization**, with a few caveats (one needs dimensional constants), to solve the **composite translational-rotational** optimization problem as well.