OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... By considering that $x_0^i = 0,\ i = 1 \ldots n$, one can simplify all partial derivatives:

$$f_\alpha^i = \tfrac{1}{2} s\alpha \big(x_1^i y_1^i u^2 + x_1^i y_2^i uv + x_1^i y_3^i uw + x_2^i y_1^i vu + x_2^i y_2^i v^2 + x_2^i y_3^i vw + x_3^i y_1^i wu + x_3^i y_2^i wv + x_3^i y_3^i w^2 \big).$$

$$f_u^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\big(2 x_1^i y_1^i u + (x_1^i y_2^i + x_2^i y_1^i) v + (x_1^i y_3^i + x_3^i y_1^i) w \big).$$

$$f_v^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\big((x_1^i y_2^i + x_2^i y_1^i) u + 2 x_2^i y_2^i v + (x_2^i y_3^i + x_3^i y_2^i) w \big).$$

$$f_w^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\big((x_1^i y_3^i + x_3^i y_1^i) u + (x_2^i y_3^i + x_3^i y_2^i) v + 2 x_3^i y_3^i w \big).$$

These partial derivatives can be written in matrix notation:

$$f_\alpha^i = \tfrac{1}{2} s\alpha \cdot [x_1^i, x_2^i, x_3^i] \cdot \begin{bmatrix} u^2 & uv & uw \\ vu & v^2 & vw \\ wu & wv & w^2 \end{bmatrix} \cdot \begin{bmatrix} y_1^i \\ y_2^i \\ y_3^i \end{bmatrix} = \tfrac{1}{2} s\alpha\, \mathbb{x}_i^T \mathbb{U} \mathbb{y}_i = \tfrac{1}{2} s\alpha\, \mathbb{x}_i^T \mathbb{u} \cdot \mathbb{u}^T \mathbb{y}_i .$$

$$f_u^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot [x_1^i, x_2^i, x_3^i] \cdot \begin{bmatrix} 2y_1^i & y_2^i & y_3^i \\ 0 & y_1^i & 0 \\ 0 & 0 & y_1^i \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot \mathbb{x}_i^T Y_1^i \mathbb{u} .$$

$$f_v^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot [x_1^i, x_2^i, x_3^i] \cdot \begin{bmatrix} y_2^i & 0 & 0 \\ y_1^i & 2y_2^i & y_3^i \\ 0 & 0 & y_2^i \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot \mathbb{x}_i^T Y_2^i \mathbb{u} .$$

$$f_w^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot [x_1^i, x_2^i, x_3^i] \cdot \begin{bmatrix} y_3^i & 0 & 0 \\ 0 & y_3^i & 0 \\ y_1^i & y_2^i & 2y_3^i \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha) \cdot \mathbb{x}_i^T Y_3^i \mathbb{u} .$$

Alternatively, one can define f_u^i, f_v^i and f_w^i as follows:

$$f_u^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\, [\, x_1^i \cdot \mathbb{y}_i^T + y_1^i \cdot \mathbb{x}_i^T \,] \cdot \mathbb{u} .$$

$$f_v^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\, [\, x_2^i \cdot \mathbb{y}_i^T + y_2^i \cdot \mathbb{x}_i^T \,] \cdot \mathbb{u} .$$

$$f_w^i = (\tfrac{1}{2} - \tfrac{1}{2} c\alpha)\, [\, x_3^i \cdot \mathbb{y}_i^T + y_3^i \cdot \mathbb{x}_i^T \,] \cdot \mathbb{u} .$$

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• **Laplacian eigenfunctions and neural networks:** ... Using these definitions of the partial derivatives (of the summands in the corresponding sums), one obtains:

• **Necessary conditions for $f = \sum_{i=1}^{n} \langle R\,y_i, x_i \rangle$ to be maximal**

$$\sum_{i=1}^{n} f_\alpha^i = \sum_{i=1}^{n} \left(\tfrac{1}{2} s\alpha \, x_i^T U y_i\right) = \tfrac{1}{2} s\alpha \sum_{i=1}^{n} \left(x_i^T U y_i\right) = 0$$

$$\Longleftrightarrow \underline{s\alpha \cdot \sum_{i=1}^{n} \left(x_i^T U y_i\right) = 0}\,.$$

$$\sum_{i=1}^{n} f_u^i = \sum_{i=1}^{n} \left(\left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) x_i^T Y_1^i \, u\right) = \left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) \sum_{i=1}^{n} \left(x_i^T Y_1^i \, u\right) = 0$$

$$\Longleftrightarrow \underline{(1 - c\alpha)\left(\sum_{i=1}^{n} x_i^T Y_1^i\right) u = 0}\,.$$

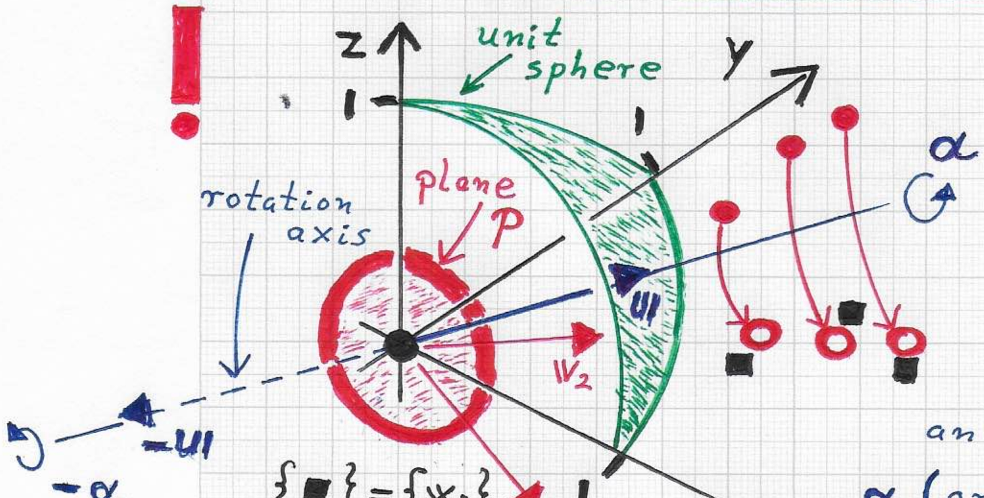
$$\sum_{i=1}^{n} f_v^i = \sum_{i=1}^{n} \left(\left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) x_i^T Y_2^i \, u\right) = \left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) \sum_{i=1}^{n} \left(x_i^T Y_2^i \, u\right) = 0$$

$$\Longleftrightarrow \underline{(1 - c\alpha)\left(\sum_{i=1}^{n} x_i^T Y_2^i\right) u = 0}\,.$$

$$\sum_{i=1}^{n} f_w^i = \sum_{i=1}^{n} \left(\left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) x_i^T Y_3^i \, u\right) = \left(\tfrac{1}{2} - \tfrac{1}{2} c\alpha\right) \sum_{i=1}^{n} \left(x_i Y_3^i \, u\right) = 0$$

$$\Longleftrightarrow \underline{(1 - c\alpha)\left(\sum_{i=1}^{n} x_i^T Y_3^i\right) u = 0}\,.$$

Normalization constraint: $\|u\|^2 = u^2 + v^2 + w^2 = 1$.

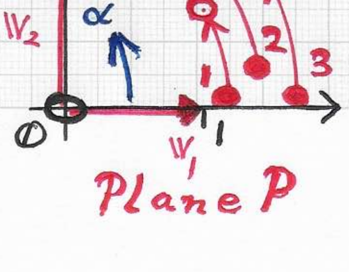


$\{■\} = \{x_i\}$
$\{●\} = \{y_i\}$
$\{○\} = \{\hat{y}_i\}$
$\langle u, v_1 \rangle = \langle u, v_2 \rangle = 0$

The left figure illustrates geometrical issues of the problem. The goal is to determine the optimal unit direction vector u (or $-u$) and optimal rotation angle α (or $-\alpha$). This vector and this angle define a rotation of a given point set $\{y_i\}_{i=1}^{n}$ such that the resulting image point set $\{\hat{y}_i\}_{i=1}^{n}$ has minimal distance to point set $\{x_i\}_{i=1}^{n}$ in a least-squares sense. ...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... The figure on the previous page also shows the plane P passing through the origin and having the vector $\mathbf{u}_1$ $(-\mathbf{u}_1)$ as its normal vector. Further, two vectors, $\mathbf{v}_1$ and $\mathbf{v}_2$, are shown; they are assumed to be orthonormal, defining a (right-handed) 2D coordinate system for points in the plane P. Considering P, $\mathbf{v}_1$ and $\mathbf{v}_2$, one can view the optimization problem as follows: If one knew the optimal rotation axis direction vector $\mathbf{u}_1$, one could orthogonally project the point sets $\{\mathbf{y}_i\}_{i=1}^n$ and $\{\mathbf{x}_i\}_{i=1}^n$ into the plane P; subsequently, one could determine the optimal rotation angle α that rotates the points resulting from the projection of $\{\mathbf{y}_i\}_{i=1}^n$ with minimal least-squares distance error to the corresponding points resulting from the projection of $\{\mathbf{x}_i\}_{i=1}^n$. This viewpoint reduces the calculation of the optimal angle α to a problem considering rotation around the origin in a/the plane P: One determines the rotation angle α such that the sum of the squared distances between the images of the points '●' (='○') and the corresponding points '■' is minimal. The left figure sketches the data in plane P.



Plane P

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• **Laplacian eigenfunctions and neural networks:...** The four necessary conditions for $f = \sum_{i=1}^{n} \langle R y_i, x_i \rangle$ to be maximal "formally" make it possible to consider $s\alpha = 0$ ($\Rightarrow \alpha = \ldots, -2\pi, -\pi, 0, \pi, 2\pi, \ldots$) and $(1 - c\alpha) = 0$ $\Leftrightarrow$ $c\alpha = 1$ ($\Rightarrow \alpha = \ldots, -4\pi, -2\pi, 0, 2\pi, 4\pi, \ldots$). Geometrical reasoning eliminates this "formal angle possibilities," and the four conditions become:

(i) $\sum_{i=1}^{n} (x_i^T U y_i) = 0$; (ii) $(\sum_{i=1}^{n} x_i^T Y_1^i) u = 0$;

(iii) $(\sum_{i=1}^{n} x_i^T Y_2^i) u = 0$; (iv) $(\sum_{i=1}^{n} x_i^T Y_3^i) u = 0$.

Further, the "3D geometrical constraint" is pointed out on page 17 (2-3-2024) as well — the normalization constraint concerning direction u:

$\|u\|^2 = u^2 + v^2 + w^2 = 1$.

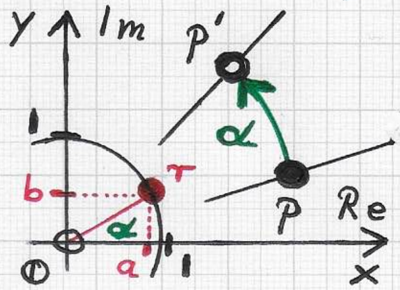
Furthermore, since the objective is the calculation of the "four-dimensional" rotation quaternion $q_{rot} = c\frac{\alpha}{2} + s\frac{\alpha}{2}\langle \hat{u}, u \rangle = c\frac{\alpha}{2} + ius\frac{\alpha}{2} + jvs\frac{\alpha}{2} + kws\frac{\alpha}{2}$, one must consider requirements concerning the norm of q_{rot}. The value of this norm is defined as $\|q_{rot}\| = c^2\frac{\alpha}{2} + u^2 s^2\frac{\alpha}{2} + v^2 s^2\frac{\alpha}{2} + w^2 s^2\frac{\alpha}{2} = \|q_{rot}\| =$ $= c^2\frac{\alpha}{2} + s^2\frac{\alpha}{2}(u^2 + v^2 + w^2)$. SINCE $c^2 + s^2 = 1$, for any value of α, THE NORMALIZATION CONSTRAINT FOR THE ROTATION QUATERNION q_{rot} IS

$\|q_{rot}\| = \cos^2\frac{\alpha}{2} + \sin^2\frac{\alpha}{2}(u^2 + v^2 + w^2) = 1$.

This constraint is important, and it must be considered in conjunction with conditions (i) – (iv), provided above, to solve the optimization problem. ...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... Rotation in 3D space represented in quaternion notation is a generalization of rotation in 2D space using "standard" complex number notation. Thus, it is helpful to briefly review the least-squares optimization problem in the 2D plane, concerned with the best-possible rotation of a point set around the origin. The



left figure illustrates the 2D case. Using complex numbers, the point $p = x + iy$ is mapped to the point $p' = x' + iy'$. The rotation is performed via the rotator $r = a + ib$, where $a = \cos\alpha$ and $b = \sin\alpha$, i.e., $a^2 + b^2 = 1$ (implying that r is a complex number "on the unit circle with center O). The image point p' is defined by the multiplication of the complex numbers r and p:

$$p' = (a+ib)\cdot(x+iy) = (ax-by) + i(ay+bx) = x' + iy'.$$

This result can also be represented by matrix multiplication, using the notation $\mathbb{p} = (x, y)^T$, $\mathbb{p}' = (x', y')^T$:

$$\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\alpha - y\sin\alpha \\ x\sin\alpha + y\cos\alpha \end{bmatrix} = \begin{bmatrix} xa - yb \\ xb + ya \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}.$$

We can now consider this optimization problem: Given point sets $\{\mathbb{p}_i\}_{i=1}^n$ and $\{\mathbb{p}_i'\}_{i=1}^n$, determine the angle α of the rotation that "maps $\mathbb{p}_i$ to $\mathbb{p}_i'$, $i = 1 \ldots n$," with minimal error.

...