OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• **Laplacian eigenfunctions and neural networks:...** Thus, we perform the three mappings in the sequence i), then ii) and finally iii). The given point is $\mathbb{P} = (x,y)^T \mathrel{\hat{=}} x + iy$, and the final result will be $\mathbb{P}' = (x', y')^T \mathrel{\hat{=}} x' + iy'$. We can now define $\mathbf{Ref_L}\,\beta \cdot \mathbb{P} = \mathbb{P}'$ using complex number notation.

i) $Rot\,{-\beta} \cdot \mathbb{P} \mathrel{\hat{=}} (c{-\beta} + is{-\beta}) \cdot (x+iy) =$

$$= (c\beta - is\beta) \cdot (x+iy) = (xc\beta + ys\beta) + i(yc\beta - xs\beta) =$$

$$= z_1 \quad \left(\mathrel{\hat{=}} \begin{pmatrix} c\beta & s\beta \\ -s\beta & c\beta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \right).$$

ii) Reflection with respect to the Re-axis (x-axis):

$$z_2 = \overline{z_1} = (xc\beta + ys\beta) - i(yc\beta - xs\beta) =$$

$$= (xc\beta + ys\beta) + i(xs\beta - yc\beta).$$

("Conjugation of a complex number $\mathrel{\hat{=}}$ Ref_x.")

iii) $Rot\,\beta \cdot z_2 \mathrel{\hat{=}} (c\beta + is\beta) \cdot \left([xc\beta + ys\beta] + i[xs\beta - yc\beta] \right) =$

$$= [xc^2\beta + yc\beta s\beta - xs^2\beta + ys\beta c\beta]$$

$$+ i[xs\beta c\beta + ys^2\beta + xc\beta s\beta - yc^2\beta] =$$

$$= [x(c^2\beta - s^2\beta) + 2ys\beta c\beta]$$

$$+ i[2xs\beta c\beta - y(c^2\beta - s^2\beta)] =$$

$$= [xc(2\beta) + ys(2\beta)] + i[xs(2\beta) - yc(2\beta)]$$

$$= (c2\beta + is2\beta) \cdot (x - iy) = (c2\beta + i2\beta) \cdot \overline{P} = P'$$

$$\left(\mathrel{\hat{=}} \begin{pmatrix} c2\beta & s2\beta \\ s2\beta & -c2\beta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = Ref_L\,\beta \cdot \mathbb{P} \right).$$

• **Note.** One can more compactly write this mapping as

$$P' = (c\beta + is\beta) \cdot \overline{\left((c{-\beta} + is{-\beta}) \cdot (x+iy) \right)}$$

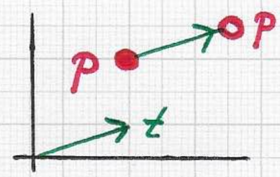
$$= \ldots = (c(2\beta) + is(2\beta)) \cdot \overline{P} \mathrel{\hat{=}} \mathbb{P}'.$$

...

OBJECT AND MATERIAL EIGENFUNCTIONS — Cont'd.

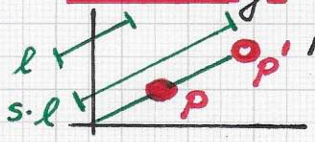
- **Laplacian eigenfunctions and neural networks:...** We can now summarize the discussed fundamental geometric transformations in complex number notation:

- **Translation:** $p = x + iy$



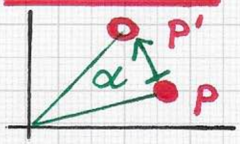
$$\mapsto p' = x + t_x + i(y + t_y) = x + iy + t_x + it_y = p + t = x' + iy'$$

- **Scaling:** $p = x + iy$



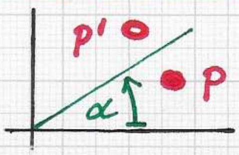
$$\mapsto p' = (s + i0)\cdot(x + iy) = sx + isy = s \cdot p = x' + iy'$$

- **Rotation:** $p = x + iy$



$$\mapsto p' = (c\alpha + is\alpha)\cdot(x + iy) = xc\alpha - ys\alpha + i(xs\alpha + yc\alpha) = x' + iy'$$

- **Reflection:** $p = x + iy$



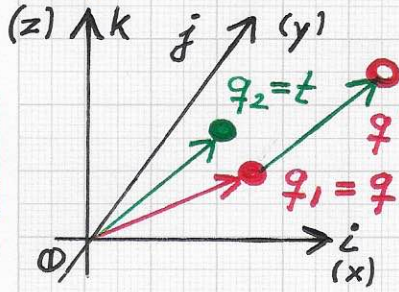
$$\mapsto p' = (c2\alpha + is2\alpha)\cdot(x - iy) = (c2\alpha + is2\alpha)\cdot\overline{p} = x' + iy'$$

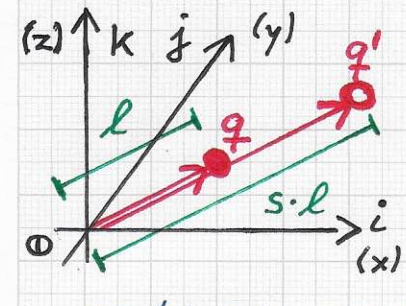
One can view these transformations as "building blocks" of a geometric algebra for the (complex) plane, allowing one to map points (complex numbers) simply by performing operations with complex numbers — and thus "eliminating" the use of matrix algebra to achieve the desired geometrical result. One can generalize this concept for mapping points in **3D** space purely algebraically.

...

OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

- **Laplacian eigenfunctions and neural networks:...** Quaternions provide the algebraic foundation for performing geometrical transformations in 3D space usually represented and achieved via equivalent matrix algebra. Again, we only consider the transformations translation, scaling, rotation and reflection. (Of course, one can also express computations like scalar/dot and vector products in quaternion algebra.) Generally, one can use the three "complex terms" of a quaternion (i, j ank terms) to represent the three coordinate components (of a positional vector) of a point in 3D space. Since addition of two quaternions q_1 and q_2 and scalar multiplication of a quaternion by a "scaling factor" s are component-wise operations, one can simply use the complex ("vector") part of a quaternion(s) to represent the geometrical transformations translation and scaling.





- **Translation** — by $\mathbf{t} = (t_x, t_y, t_z)^T \hat{=} 0 + it_x + jt_y + kt_z$

$$\mathbf{p}' = (x', y', z')^T = \mathbf{p} + \mathbf{t} = (x, y, z)^T + (t_x, t_y, t_z)^T = (x + t_x, y + t_y, z + t_z)^T$$

$$\hat{=}\ q' = 0 + ix' + jy' + kz' = q + t = (0 + ix + jy + kz) + (0 + it_x + jt_y + kt_z)$$

$$= 0 + (x + t_x)i + (y + t_y)j + (z + t_z)k.$$

(See above figures.)

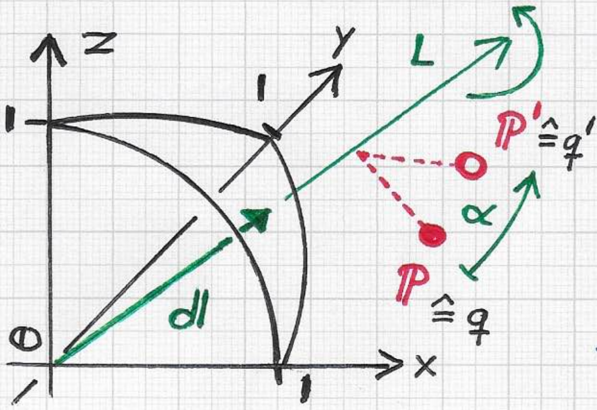
...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks: ... • Scaling – by $s \hat{=} s + 0i + 0j + 0k$

$\mathbb{p}' = (x', y')^T = s \cdot \mathbb{p} = s(x,y)^T = (sx, sy)^T$

$\hat{=}\; q' = 0 + ix' + jy' + kz' = s \cdot q = 0 + isx + jsy + ksz$.



The left figure sketches the scenario of rotating a point $\mathbb{p}$ by an angle α around an axis L, passing through the origin and oriented, defined by a unit, normalized direction vector $\mathbf{dl} = (d_x, d_y, d_z)^T$, $\|\mathbf{dl}\| = 1$. Using quaternion representation, this rotation can be compactly expressed.

• Rotation – by angle α around axis L

$\mathbb{p}' = (x', y', z')^T = M_3 \cdot M_2 \cdot M_1 \cdot \mathbb{p}$, where

M_1 = matrix Mapping Vector_d To Vector_001,

M_2 = matrix Rotating By Alpha Around Zaxis,

$M_3 = M^{-1}$ (= inverse of matrix M_1)

$$\hat{=}\; q' = a' + ib' + jc' + kd' = q_{rot} \cdot q \cdot \overline{q_{rot}} =$$
$$= \left(\cos(\tfrac{\alpha}{2}) + (id_x + jd_y + kd_z)\cdot \sin(\tfrac{\alpha}{2})\right)$$
$$\cdot \left(0 + ix + jy + kz\right)$$
$$\cdot \left(\cos(\tfrac{\alpha}{2}) - (id_x + jd_y + kd_z)\cdot \sin(\tfrac{\alpha}{2})\right)$$
$$= \left(c\tfrac{\alpha}{2} + \langle \mathbf{i}, \mathbf{dl}\rangle s\tfrac{\alpha}{2}\right) \cdot \left(\langle \mathbf{i}, \mathbf{x}\rangle\right) \cdot \left(c\tfrac{\alpha}{2} - \langle \mathbf{i}, \mathbf{dl}\rangle s\tfrac{\alpha}{2}\right),$$

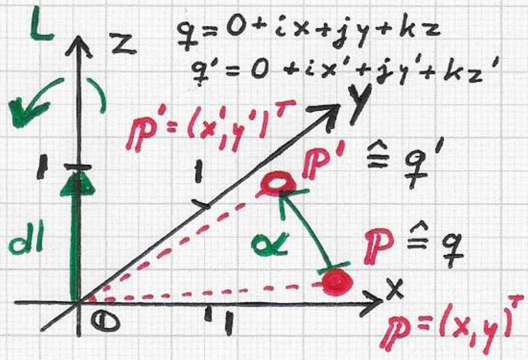
where $\mathbf{i} = (i, j, k)$, $\mathbf{dl} = (d_x, d_y, d_z)$ and $\mathbf{x} = (x, y, z)$.

(• Note. One can use this representation also to describe a rotation in the 2D plane by using the direction vector $\mathbf{dl} = (0,0,1)^T$ and point $\mathbb{p} = (x, y, 0)^T$.)

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

- Laplacian eigenfunctions and neural networks:...



Rotation in xy-plane as rotation in xyz-space.

We can understand a rotation around the origin by an angle α in the 2D xy-plane as a rotation around the oriented z-axis by an angle α in 3D xyz-space, with $dl = (0,0,1)^T$, see left figure. Using quaternion representation as defined on the previous page, this rotation in the plane is:

$$q' = q_{rot} \cdot q \cdot \overline{q_{rot}} = \left(c\tfrac{\alpha}{2} + \langle u, dl\rangle s\tfrac{\alpha}{2}\right) \cdot \langle u, x\rangle \cdot \left(c\tfrac{\alpha}{2} - \langle u, dl\rangle s\tfrac{\alpha}{2}\right) =$$

(where $u = (i, j, k)$, $dl = (0,0,1)$, $x = (x, y, 0)$)

$$= \left(c\tfrac{\alpha}{2} + k s\tfrac{\alpha}{2}\right) \cdot (ix + jy) \cdot \left(c\tfrac{\alpha}{2} - k s\tfrac{\alpha}{2}\right) =$$

$$= \left(c\tfrac{\alpha}{2} + k s\tfrac{\alpha}{2}\right) \cdot \left(ixc\tfrac{\alpha}{2} - ikxs\tfrac{\alpha}{2} + jyc\tfrac{\alpha}{2} - jkys\tfrac{\alpha}{2}\right) =$$

$$= ixc^2\tfrac{\alpha}{2} - ikxc\tfrac{\alpha}{2}s\tfrac{\alpha}{2} + jyc^2\tfrac{\alpha}{2} - jkyc\tfrac{\alpha}{2}s\tfrac{\alpha}{2}$$
$$+ kixs\tfrac{\alpha}{2}c\tfrac{\alpha}{2} - kikxs^2\tfrac{\alpha}{2} + kjys\tfrac{\alpha}{2}c\tfrac{\alpha}{2} - kjkys^2\tfrac{\alpha}{2} =$$

$$= ixc^2\tfrac{\alpha}{2} + jxc\tfrac{\alpha}{2}s\tfrac{\alpha}{2} + jyc^2\tfrac{\alpha}{2} - iyc\tfrac{\alpha}{2}s\tfrac{\alpha}{2}$$
$$+ jxs\tfrac{\alpha}{2}c\tfrac{\alpha}{2} - ixs^2\tfrac{\alpha}{2} - iys\tfrac{\alpha}{2}c\tfrac{\alpha}{2} - jys^2\tfrac{\alpha}{2} =$$

$$= (ix + jy)c^2\tfrac{\alpha}{2} - (ix + jy)s^2\tfrac{\alpha}{2} + 2(jx - iy)s\tfrac{\alpha}{2}c\tfrac{\alpha}{2} =$$

$$= (ix + jy)\underbrace{\left(c^2\tfrac{\alpha}{2} - s^2\tfrac{\alpha}{2}\right)}_{=\cos\alpha} + (jx - iy)\underbrace{2s\tfrac{\alpha}{2}c\tfrac{\alpha}{2}}_{=\sin\alpha}$$

$$= (ix + jy)c\alpha + (jx - iy)s\alpha$$

$$= i(xc\alpha - ys\alpha) + j(xs\alpha + yc\alpha)$$

$$= 0 + i(xc\alpha - ys\alpha) + j(xs\alpha + yc\alpha) + k \cdot 0 = a' + ib' + jc' + kd'$$

$$= ix' + jy' \mathrel{\hat{=}} \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} c\alpha & -s\alpha \\ s\alpha & c\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

Thus, one can use "standard" complex numbers or quaternions to represent rotations in the plane. ...