Scalar product. The scalar product is a fundamental operator on two vectors, providing us with a scalar quantity. It allows us to conveniently compute projections and is widely used in all our computations. The scalar product operator is also called the "dot product" or "inner product."

The definition of the scalar product is as follows. Given two vectors $\mathbf{v}$ and $\mathbf{w}$ and the angle θ between them,

$$\langle \mathbf{v}, \mathbf{w} \rangle = \|\mathbf{v}\| \cdot \|\mathbf{w}\| \cdot \cos\theta \,.$$

Observe Figure 1.8, where ℓ denotes the length of the projection of $\mathbf{w}$ onto $\mathbf{v}$. From basic trigonometry, we have $\cos\theta = \frac{\ell}{\|\mathbf{w}\|}$, which leads to

$$\ell = \frac{\langle \mathbf{v}, \mathbf{w} \rangle}{\|\mathbf{v}\|},$$

a very useful application of the dot product operator, as we demonstrated in Equation (1.1).

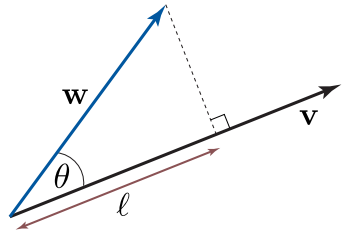

Figure 1.8: Computing $\cos\theta = \frac{\ell}{\|\mathbf{w}\|}$.

The dot product of two vectors $\langle \mathbf{v}, \mathbf{w} \rangle$ can be computed explicitly given a coordinate representation of the vectors (more on this later in Chapter 2): given $\mathbf{v} = (x_\mathbf{v}, y_\mathbf{v})$ and $\mathbf{w} = (x_\mathbf{w}, y_\mathbf{w})$, we have $\langle \mathbf{v}, \mathbf{w} \rangle = x_\mathbf{v} x_\mathbf{w} + y_\mathbf{v} y_\mathbf{w}$. In 2D we have a useful property that if $\langle \mathbf{v}, \mathbf{w} \rangle = 0$, then the vectors are perpendicular, and we can compute the perpendicular vector of $\mathbf{v} = (x_\mathbf{v}, y_\mathbf{v})$ as $\mathbf{v}^\perp = (-y_\mathbf{v}, x_\mathbf{v})$.

Parametric representation of a line. A parametric representation of a line is given by $l(t) = \mathbf{p} = \mathbf{p}_0 + t\mathbf{v}$, where the line origin is at $\mathbf{p}_0$, where $t = 0$ (see Figure 1.9); the parameter value is any real number $t \in (-\infty, \infty)$. When we refer to a ray, rather than a line, the forward direction of the ray is where $t > 0$.

With a parametric representation of a line, we can derive the distance between a point $\mathbf{q}$ and a line $l(t) = \mathbf{p}_0 + t\mathbf{v}$. As shown in

To extract the α_i, we can use the linearity of the space and form inner products that give us

$$\langle \mathbf{a}, \mathbf{v}_j \rangle = \sum_{i=1}^{n} \alpha_i \langle \mathbf{v}_i, \mathbf{v}_j \rangle \,,$$

and thus we have

$$\begin{aligned}
\langle \mathbf{a}, \mathbf{v}_1 \rangle &= \alpha_1 \langle \mathbf{v}_1, \mathbf{v}_1 \rangle + \alpha_2 \langle \mathbf{v}_2, \mathbf{v}_1 \rangle + \alpha_3 \langle \mathbf{v}_3, \mathbf{v}_1 \rangle \,; \\
\langle \mathbf{a}, \mathbf{v}_2 \rangle &= \alpha_1 \langle \mathbf{v}_1, \mathbf{v}_2 \rangle + \alpha_2 \langle \mathbf{v}_2, \mathbf{v}_2 \rangle + \alpha_3 \langle \mathbf{v}_3, \mathbf{v}_2 \rangle \,; \\
\langle \mathbf{a}, \mathbf{v}_3 \rangle &= \alpha_1 \langle \mathbf{v}_1, \mathbf{v}_3 \rangle + \alpha_2 \langle \mathbf{v}_2, \mathbf{v}_3 \rangle + \alpha_3 \langle \mathbf{v}_3, \mathbf{v}_3 \rangle \,.
\end{aligned}$$

In other words, we have a linear system of three equations and three unknowns:

$$\begin{bmatrix} \langle \mathbf{v}_1, \mathbf{v}_1 \rangle & \langle \mathbf{v}_1, \mathbf{v}_2 \rangle & \langle \mathbf{v}_1, \mathbf{v}_3 \rangle \\ \langle \mathbf{v}_2, \mathbf{v}_1 \rangle & \langle \mathbf{v}_2, \mathbf{v}_2 \rangle & \langle \mathbf{v}_2, \mathbf{v}_3 \rangle \\ \langle \mathbf{v}_3, \mathbf{v}_1 \rangle & \langle \mathbf{v}_3, \mathbf{v}_2 \rangle & \langle \mathbf{v}_3, \mathbf{v}_3 \rangle \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} \langle \mathbf{a}, \mathbf{v}_1 \rangle \\ \langle \mathbf{a}, \mathbf{v}_2 \rangle \\ \langle \mathbf{a}, \mathbf{v}_3 \rangle \end{bmatrix} .$$

Here is where orthogonality may come in handy. If the basis is orthogonal, then $\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \left[\begin{smallmatrix} 1 & i=j \\ 0 & i \neq j \end{smallmatrix} \right]$, and we get the trivial identity matrix and all the above collapses into the simple form:

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} \langle \mathbf{a}, \mathbf{v}_1 \rangle \\ \langle \mathbf{a}, \mathbf{v}_2 \rangle \\ \langle \mathbf{a}, \mathbf{v}_3 \rangle \end{bmatrix} .$$

The simplicity of this form is particularly attractive because no linear system needs to be solved (since the matrix to be inverted is the identity matrix). But orthogonality is important not only because it may simplify computations; it is also essential for numerical stability of computations, as it avoids an accumulation of roundoff errors.

Homogeneous coordinates

Earlier in this chapter, we claimed that a translation is not a linear operation. We attributed it to the fact that a translation operator T might translate the origin: $T(\mathbf{0}) \neq \mathbf{0}$. This is one of the reasons why *homogeneous coordinates* are commonly used in computer graphics. In homogeneous coordinates, translation is

If the rank of A is smaller than n, then A is singular and it maps the entire space $\mathbb{R}^n$ onto some subspace, like a plane (so A is some sort of a projection).

Earlier we discussed computing the principal directions of a set of points $\mathbf{x}_i$, or in fact the vectors $\mathbf{y}_i$ (which are $\mathbf{x}_i$ shifted by the center of mass $\mathbf{m}$): $\mathbf{y}_i = \mathbf{x}_i - \mathbf{m}$. Let Y be the matrix whose columns are the vectors $\mathbf{y}_i$:

$$Y = \begin{bmatrix} | & | & & | \\ \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n \\ | & | & & | \end{bmatrix}.$$

Suppose we have the SVD of Y: $Y = U\Sigma V^\top$. Then the scatter matrix $S = YY^\top$ simplifies to

$$\begin{aligned} YY^\top &= U\Sigma V^\top (U\Sigma V^\top)^\top \\ &= U\Sigma V^\top V \Sigma^\top U^\top \\ &= U(\Sigma\Sigma^\top)U^\top. \end{aligned}$$

Thus, the column vectors of U are the principal components of the data set, and commonly they are sorted by the size of the singular values of Y.

Now, we can go back to the question raised in Chapter 2 and show how SVD can solve it: Given two objects with corresponding landmarks (shown in Figure 4.14 (left)), how can we find a rigid transformation that aligns them (Figure 4.14 (right))? When the objects are aligned, the lengths of the line segments connecting the landmarks are small. Therefore, we can solve a least-squares problem. Let $\mathbf{p}_i$ and $\mathbf{q}_i$ be the corresponding sets of points. We seek a translation vector $\mathbf{t}$ and a rotation matrix R so that $\sum_{i=1}^n \|\mathbf{p}_i - (R\,\mathbf{q}_i + \mathbf{t})\|^2$ is minimized.

It turns out that we can solve for the translation and rotation separately. If $(R, \mathbf{t})$ is the optimal transformation, then the points $\{\mathbf{p}_i\}$ and $\{R\,\mathbf{q}_i + \mathbf{t}\}$ have the same centers of mass. ~~To see that, let $\mathbf{p} = \frac{1}{n}\sum_{i=1}^n \mathbf{p}_i$ and $\mathbf{q} = \frac{1}{n}\sum_{i=1}^n \mathbf{q}_i$. Given optimal R and $\mathbf{t}$,~~

~~$$\mathbf{p} = \frac{1}{n}\sum_{i=1}^n (R\,\mathbf{q}_i + \mathbf{t}) = R\left(\frac{1}{n}\sum_{i=1}^n \mathbf{q}_i\right) + \mathbf{t} = R\,\mathbf{q} + \mathbf{t},$$~~

~~and $\mathbf{t} = \mathbf{p} - R\,\mathbf{q}$. The~~ This can be shown by differentiating our least-squares objective with respect to $\mathbf{t}$.

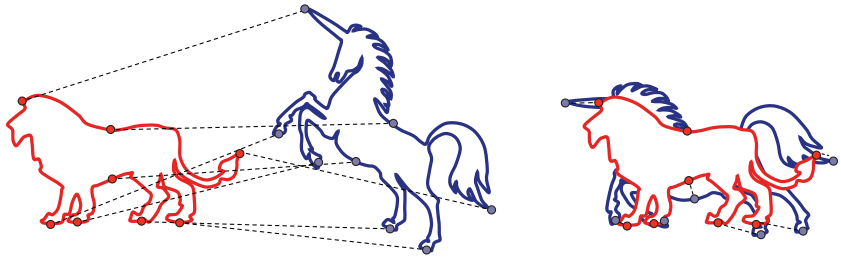

Figure 4.14: Aligning two shapes in correspondence by a rigid transformation.

To find R, let us assume that the corresponding points have been shifted so that their centers of mass align. Now we want to find R that minimizes

$$\sum_{i=1}^{n} \|\mathbf{p}_i - R\mathbf{q}_i\|^2.$$

Let

$$H = \sum_{i=1}^{n} \mathbf{q}_i \mathbf{p}_i^\top.$$

Given the SVD of $H = U\Sigma V^\top$, the optimal orthogonal transformation is $R = VU^\top$. Below we see why.

Given the orthogonality of R, we have $R^\top R = I$ and hence

$$\begin{aligned}
\sum_{i=1}^{n} \|\mathbf{p}_i - R\,\mathbf{q}_i\|^2 &= \sum_{i=1}^{n} (\mathbf{p}_i - R\,\mathbf{q}_i)^\top (\mathbf{p}_i - R\,\mathbf{q}_i) \\
&= \sum_{i=1}^{n} \mathbf{p}_i^\top \mathbf{p}_i - \mathbf{p}_i^\top R\,\mathbf{q}_i - \mathbf{q}_i^\top R^\top \mathbf{p}_i + \mathbf{q}_i^\top R^\top R\,\mathbf{q}_i \\
&= \sum_{i=1}^{n} \mathbf{p}_i^\top \mathbf{p}_i - \mathbf{p}_i^\top R\,\mathbf{q}_i - \mathbf{q}_i^\top R^\top \mathbf{p}_i + \mathbf{q}_i^\top \mathbf{q}_i\,.
\end{aligned}$$

The first and last terms, $\mathbf{p}_i^\top \mathbf{p}_i$ and $\mathbf{q}_i^\top \mathbf{q}_i$, do not depend on R, so we can ignore them in the minimization of $\sum_{i=1}^{n} \|\mathbf{p}_i - R\,\mathbf{q}_i\|^2$. Thus, the minimization reduces to

$$\min_R \sum_{i=1}^{n} (-\mathbf{p}_i^\top R\,\mathbf{q}_i - \mathbf{q}_i^\top R^\top \mathbf{p}_i) = \max_R \sum_{i=1}^{n} (\mathbf{p}_i^\top R\,\mathbf{q}_i + \mathbf{q}_i^\top R^\top \mathbf{p}_i)\,.$$

Since the second term $\mathbf{q}_i^\top R^\top \mathbf{p}_i$ is a scalar, we have

$$\mathbf{p}_i^\top R\,\mathbf{q}_i = (\mathbf{p}_i^\top R\,\mathbf{q}_i)^\top = \mathbf{q}_i^\top R^\top \mathbf{p}_i,$$

which implies that we are looking for

$$\operatorname*{argmax}_R \sum_{i=1}^{n} 2\mathbf{p}_i^\top R\,\mathbf{q}_i = \operatorname*{argmax}_R \sum_{i=1}^{n} \mathbf{p}_i^\top R\,\mathbf{q}_i\,.$$

Simplifying further,

$$\sum_{i=1}^{n} \mathbf{p}_i^\top R\,\mathbf{q}_i = \text{Trace}\left(\sum_{i=1}^{n} R\,\mathbf{q}_i\mathbf{p}_i^\top\right) = \text{Trace}\left(R\sum_{i=1}^{n} \mathbf{q}_i\mathbf{p}_i^\top\right),$$

where $\text{Trace}(A) = \sum_{i=1}^{n} A_{ii}$.

Hence, we want to find R that maximizes $\text{Trace}(RH)$. It is known that if M is symmetric positive definite (all eigenvalues of M are positive) and B is any orthogonal matrix, then

$$\text{Trace}(M) \geq \text{Trace}(BM)\,.$$

Thus, let us find R so that RH is symmetric positive definite. Then we know for sure that $\text{Trace}(RH)$ is maximal. If $H = U\Sigma V^\top$ is the SVD, we define $R = VU^\top$. Now let us check RH:

$$RH = (VU^\top)(U\Sigma V^\top) = V\Sigma V^\top,$$

which is a symmetric matrix and its eigenvalues are positive, meaning that RH is symmetric positive definite. If you want to go deeper into the algebra of this problem, refer to our technical note "Least-Squares Rigid Motion Using SVD",
`http://igl.ethz.ch/projects/ARAP/svd_rot.pdf`

Practical computation of principal components. Computing the SVD is expensive, and we always need to pay attention to the dimensions of the matrix. In many applications we need to compute the principal components for very large matrices, and that can be extremely computationally expensive or even infeasible. Suppose, for example, that each vector represents the pixels of an image: then the vector length is approximately $16K$ for a rather small-size image of 128×128 pixels, and the scatter matrix is a huge matrix of dimensions $16K \times 16K$. However, the rank