ORTHOGONALITY · GRAM-SCHMIDT

The same algorithm across settings

The procedure never changes — subtract projections, optionally normalise. What changes from row to row is only the inner product being used, and with it what the word "orthogonal" turns out to mean.

6 SETTINGS

THE DOT PRODUCT ON $\mathbb{R}^n$ — 3

1 **Orthogonal basis**

input independent

$$\mathbf{u}_j = \mathbf{v}_j - \sum_{i<j} \operatorname{proj}_{\mathbf{u}_i}(\mathbf{v}_j)$$

Each vector loses everything it shares with the ones already processed, leaving the perpendicular remainder. The span is preserved at every prefix, so the output describes the same subspace as the input — differently oriented, not differently sized.

2 **Orthonormal basis**

one extra division per vector

$$\mathbf{q}_j = \mathbf{u}_j/\|\mathbf{u}_j\|$$

Normalisation is a separate step and strictly optional — orthogonality is already achieved. It is worth doing because projection onto an orthonormal set is a bare dot product, with no denominator to carry.

3 **QR decomposition**

A of full column rank

$A = QR$, R upper triangular

Run the algorithm on the columns of A and the outputs are the columns of Q; the projection coefficients discarded along the way are exactly the entries of R. The factorization is the algorithm with its bookkeeping written down — see **QR**.

OTHER INNER PRODUCTS — 3

4 **Polynomials**

applied to $\{1, x, x^2, \ldots\}$

$$\langle p, q\rangle = \int_{-1}^{1} p(x)q(x)\,dx$$

The monomials are independent but not orthogonal under this product. Orthogonalizing them produces the Legendre polynomials — which is to say Legendre polynomials are not a discovery so much as what Gram–Schmidt returns when handed the obvious basis.

5 **Functions**

sines and cosines

$$\langle f, g\rangle = \int_{0}^{2\pi} f(x)g(x)\,dx$$

Here the algorithm has almost nothing to do: distinct frequencies are already orthogonal under this product, so it only normalises. That is the fact the Fourier basis rests on, and the reason a Fourier coefficient is a single integral rather than a linear system.

6 **Any inner product space**

finite-dimensional, any $\langle\cdot,\cdot\rangle$

orthonormal $\{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$

The general statement the rows above are instances of. Because the algorithm terminates and uses only the axioms, it proves the existence of an orthonormal basis rather than assuming it — a constructive proof, not an abstract one.

The algorithm only ever calls two things: the inner product and scalar multiplication. It never inspects what a vector is, which is why the same code orthogonalizes tuples, polynomials and functions without modification — and why it constitutes a constructive proof that every finite-dimensional **inner product space** has an orthonormal basis.